

# COINS BI Toolset User Guide

---

Prepared by:	Documentation Team
Prepared for:	Learning Resources
Module:	Business Intelligence
Date:	2016
Document Ref:	
Version:	xx.xx

Construction Industry Solutions Ltd.  
11 St. Laurence Way  
SL1 2EA



Copyright 2016 Construction Industry Solutions Ltd.. All rights reserved.

Information in this document is subject to change without notice. The software described in this document is furnished under a license agreement or nondisclosure agreement. The software may be used or copied only in accordance with the terms of those agreements. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or any means electronic or mechanical, including photocopying and recording for any purpose other than the purchaser's personal use without the written permission of Construction Industry Solutions Ltd..

Construction Industry Solutions Ltd.  
11 St. Laurence Way  
SL1 2EA

**THIS USER GUIDE WAS CREATED USING MADCAP FLARE.**



# 1 Contents

<b>1 Contents</b> .....	<b>3</b>
<b>2 Database Overview</b> .....	<b>21</b>
2.1 Relational Database Model .....	22
2.2 Tables and Modules .....	23
2.2.1 Database Structure .....	24
2.2.2 Company Specific Modules .....	25
2.2.3 Process Specific Modules : .....	25
2.3 Summary and Detail Tables .....	27
2.4 Open Items .....	28
2.5 Indexes .....	29
2.6 Record Service Procedures .....	30
2.7 Read Only Fields .....	33
2.7.1 Exposing RO Fields .....	35
2.8 Create and Maintain Lookups .....	37
<b>3 OA Query Language</b> .....	<b>39</b>
3.1 FOR EACH .....	40
3.2 WHERE .....	41
3.3 Joining Tables .....	43
3.3.1 EACH .....	43
3.3.2 FIRST .....	43
3.3.3 LAST .....	43
3.3.4 OF .....	44
3.4 Curly Braces .....	45
3.5 Outer-Join .....	46



3.6	Buffers .....	48
3.7	Calculation Programs .....	49
3.7.1	Creating Calculation Programs .....	49
3.7.2	Calling Calculation Programs .....	49
<b>4</b>	<b>OA Calculation Syntax .....</b>	<b>51</b>
4.1	Finding the Period Number from a Date .....	52
4.2	syuquery aggregate .....	53
4.3	Performing Calculations on Numerical Values .....	54
4.3.1	A Simple Count .....	55
4.3.2	String\$ .....	56
4.3.3	IF .....	57
4.3.4	RANGE .....	60
4.3.5	Limit .....	62
4.3.6	Max .....	63
4.3.7	Min .....	64
4.3.8	Sum .....	65
4.3.9	non-zero .....	65
4.3.10	entry .....	66
4.3.11	truncate .....	67
4.3.12	round .....	68
4.4	String Calculations .....	69
4.4.1	INTEGER .....	70
4.4.2	DECIMAL .....	71
4.4.3	INLIST .....	72
4.4.4	CAN-DO .....	73



4.4.5 MASK .....	74
4.4.6 ENTRY\$ .....	75
4.4.7 RIGHT\$ .....	76
4.4.8 LEFT\$ .....	77
4.4.9 SUBSTRING\$ .....	78
4.4.10LENGTH\$ .....	79
4.4.11REPLACE\$ .....	80
4.4.12CAPS\$ .....	81
4.4.13TRIM\$ .....	82
4.4.14RIGHT-TRIM\$ .....	84
4.4.15INDEX .....	85
4.4.16R-INDEX .....	86
4.4.17NUM-ENTRIES .....	87
4.4.18LOOKUP .....	88
4.5 Time & Date Calculations .....	89
4.5.1 date .....	90
4.5.2 datestring .....	91
4.5.3 weekday .....	92
4.5.4 day .....	93
4.5.5 month .....	94
4.5.6 year .....	95
4.5.7 time .....	96
4.5.8 Weekdays .....	97
<b>5 OA and BI Utilities .....</b>	<b>98</b>
5.1 Database Enquiry .....	99

5.2	Access to the Database Enquiry and Tables .....	101
5.3	Query Editor .....	103
5.3.1	Search and Replace .....	106
5.3.2	Exporting from the Query Editor .....	108
5.3.3	Saving Queries in Query Editor (10.27) .....	108
5.3.4	Creating Datasets from the Query Editor .....	110
5.4	Calculation Editor .....	111
5.5	Object Enquiry .....	112
<b>6</b>	<b>End User Reporting .....</b>	<b>114</b>
6.1	OA Report Writer .....	115
6.1.1	Key features .....	115
6.1.2	Key Functionality: .....	115
6.1.3	Granting Access to Report Writer and Report Runner .....	116
6.1.3.1	Function Security .....	117
	Report Runner .....	118
6.1.4	Report Data Sources .....	120
6.1.5	Report Data Source Definitions .....	121
6.1.6	Report Writer - Creating a New Report .....	123
6.1.7	Selecting the fields to print on the report .....	127
6.1.7.1	New Reports .....	127
6.1.8	Sorting, Sub-totalling, Page Breaks and Summary Levels .....	127
6.1.9	Previewing or exporting the data .....	128
6.1.10	Running a Report .....	128
6.1.11	Publishing Report Writer Reports .....	130
6.1.11.1	Publishing a report to Report Runner .....	131

6.1.11.2 Convert to Function .....	132
Amending Reports that have been converted to Functions .....	132
6.1.11.3 Convert to Report Designer .....	134
Amending Reports that have been converted to OA Designer .....	135
6.1.12 Read Only (Calculated) Fields .....	136
6.1.12.1 Using Parameter driven RO_ fields in Report Writer .....	137
6.1.12.2 Exposing RO Fields .....	138
6.1.12.3 Using RO Fields directly in a calculation .....	140
6.2 Semantic Layer Exercises - Overview .....	141
6.2.1 Exercise - A Simple Listing Report .....	141
6.2.1.1 Build the Data Set .....	141
6.2.1.2 Test the Dataset .....	144
6.2.1.3 Make the Dataset available to the Reporting Tools .....	146
6.2.2 Using the Dataset in a Report Writer Report .....	148
6.2.3 Using the Dataset in an OA Designer Report .....	151
6.2.4 Exercise - Adding Calculated Fields to a Collection .....	155
6.2.4.1 Add the fields .....	155
6.2.4.2 Add the fields to the Dataset .....	159
6.2.4.3 Test the Dataset .....	161
6.2.5 Exercise - Adding Fields to a Collection Table .....	162
6.2.5.1 Add a new Token .....	163
6.2.6 Exercise - Adding fields to a Collection .....	168
6.2.6.1 Add the fields .....	168
6.2.6.2 Add the fields to the Dataset .....	172
6.2.6.3 Test the Dataset .....	173

6.2.7 Exercise - Summaries .....	175
6.2.7.1 Using the summary data in a report .....	179
6.3 Report Builder .....	180
6.3.1 Initial Setup .....	181
6.3.1.1 Browser Version .....	181
6.3.1.2 Report Builder System Parameters .....	182
6.3.1.3 Report Data Sources – Pre-Built Datasets .....	183
6.3.1.4 Security .....	184
6.3.1.5 Query Results Licence .....	185
6.3.2 Report Builder .....	186
6.3.2.1 Canvas .....	187
6.3.2.2 Button and Options Bar .....	188
Report Actions .....	188
Field Formatting .....	188
Canvas View .....	189
Actions .....	189
Run .....	189
6.3.2.3 Opening an Existing Report .....	191
6.3.2.4 Creating a New Report .....	192
Fields .....	192
6.3.2.5 Data Source .....	194
Existing Data Source .....	194
Custom Data Source .....	195
Data Frame .....	196
6.3.2.6 Adding Fields .....	198

6.3.2.7 Groups .....	202
Fields .....	202
6.3.2.8 Filters .....	204
6.3.2.9 Run Report .....	207
6.3.2.10 Refining the Report - Fields .....	209
6.3.2.11 Field Positioning .....	212
6.3.2.12 Text, Count, Calculations .....	213
6.3.2.13 Dataset Field Update .....	215
6.3.2.14 Adding Fields to Datasets .....	216
6.3.2.15 Report Builder List .....	219
Export Report Builder Definition .....	220
Import Report Builder Definition .....	222
6.4 Dynamic Queries .....	224
6.4.1 Dynamic Query Definition .....	225
6.4.2 Dynamic Enquiry .....	228
6.4.3 Report .....	231
6.4.4 Dynamic Grouping .....	235
6.4.5 Report Builder .....	236
<b>7 OA Designer - Overview .....</b>	<b>239</b>
7.1 Designer key features .....	239
7.2 Functions and Sections .....	240
7.3 Function Naming .....	241
7.4 Function Parameters .....	242
7.4.1 stn_code .....	243
7.4.2 rtn_code .....	244

7.4.3 Exptype .....	245
7.4.4 noInfo .....	246
7.4.5 ExportType .....	247
7.4.6 Sidemenuhelp .....	248
7.4.7 Passing Variables .....	249
7.5 URL Parameters .....	250
7.5.1 Helpmode .....	251
7.6 Forms Service Procedures and Report Selection Generates .....	252
7.6.1 Bill of Quantities - BQFREP .....	253
7.6.2 Cash Book - CBFREP .....	254
7.6.3 Central Repository - CIFREP .....	255
7.6.4 Company - COFREP .....	256
7.6.5 Credit Control - CNNFREP .....	257
7.6.6 Contract Sales - CSFREP .....	258
7.6.7 Fixed Assets - FAFREP .....	259
7.6.8 Facilities Management - FMFREP .....	260
7.6.9 General Ledger - GLFREP .....	261
7.7 Page Designer .....	262
7.7.1 Page Header .....	263
7.7.2 Page Header Tab .....	264
7.7.3 Page Body Tab .....	267
7.7.4 Page Script Tab .....	269
7.7.5 Page Forms Tab .....	270
7.7.5.1 Adding a Page Form .....	271
7.7.5.2 Page Forms - Add .....	274



7.7.5.3 Page Forms - Body .....	275
7.7.5.4 Page Forms - Body Detail .....	276
7.7.5.5 Page Forms - Body Selector .....	277
Body Selector Example .....	277
Populating body selectors from generic lookup tables .....	278
Syntax .....	279
7.7.5.6 Page Forms - Body Span .....	281
7.7.5.7 Page Forms - Body Update .....	282
7.7.5.8 Page Forms - Context .....	283
7.7.5.9 Page Forms - Detail .....	284
7.7.5.10 Page Forms - Footer .....	285
7.7.5.11 Page Forms - Header .....	286
7.7.5.12 Page Forms - Multiple Update .....	287
7.7.5.13 Page Forms - Record Header .....	288
7.7.5.14 Page Forms - Totals .....	289
7.7.5.15 Page Forms - Update .....	290
7.7.6 Page Fields Tab .....	291
7.7.6.1 Page Fields - View As Options .....	298
blank .....	298
Blank When Zero .....	298
Checkbox .....	298
Code with tooltip .....	298
Combo .....	298
Disabled Field .....	299
Disabled with Lookup .....	299



Editor .....	300
Email .....	300
File .....	300
Grid Element .....	300
Inline Frame .....	300
Link .....	301
List Frame .....	301
Multiple Selection .....	301
No Break .....	302
Ordered List .....	302
Password .....	303
Picture .....	303
Preformatted .....	304
Radio Set .....	304
Selection .....	304
Sorting Combos, Selection List, Multi Selects and Ordered Lists .....	305
7.7.7 Named Filters .....	306
7.7.7.1 Replacing body query .....	306
7.7.7.2 Linked Named Filters .....	306
7.7.7.3 Multiple Named Filters .....	306
7.7.7.4 Named Filter Query Conditions .....	307
7.7.7.5 Using Named Filters to define what fields are built on a form	307
Parameters: .....	307
7.7.8 Browse Filters .....	308
7.7.8.1 Parameters: .....	308

7.8 Report Designer .....	309
7.8.1 Key Features .....	309
7.8.2 Page Summary .....	310
7.8.3 Report Designer_Using RS Fields .....	311
7.8.4 Report Designer_Report Forms .....	312
7.8.5 Headers and Footers .....	313
7.8.6 Default Report Labels .....	314
7.9 Data Display Colour Ranges .....	315
7.10 Alternate Line Shading on Forms and Reports .....	316
7.10.1 Prerequisites .....	317
7.10.2 Task Summary .....	318
7.10.3 Create report field styles .....	319
7.10.4 Applying to forms .....	321
7.10.5 Applying to reports .....	325
7.11 Queries on Purchase Orders .....	329
7.11.1 po_line v po_item .....	329
7.12 Calculations Overview .....	331
7.12.1 Mathematical Functions .....	333
7.12.2 Calculation Field Rules .....	334
7.12.3 Variables .....	335
7.12.4 Debug .....	337
7.12.5 Curly Braces .....	339
7.13 Datasets - Overview .....	340
7.13.1 Creating a Data Set .....	341
7.13.1.1 Data Set Fields .....	342

7.13.2	Best Practice .....	344
7.13.3	Granting Access to Data Sets for Other Users. ....	346
7.13.4	Rules for Keys .....	347
7.13.5	Cross Modular Reporting .....	348
7.13.6	Using Data Sets in the Query Editor .....	349
7.13.7	Using the Data Set in Queries .....	350
7.13.7.1	In-Line Reports .....	351
7.13.8	Report Pre-Processing (Syuds) .....	353
7.13.8.1	Syuds.Calc .....	354
	syuds.calc example .....	354
7.13.8.2	Syuds.Debug .....	356
7.13.8.3	Syuds.Delete .....	358
7.13.8.4	Syuds.Filter .....	359
7.13.8.5	Syuds.GroupQuery .....	360
7.13.8.6	Syuds.Merge .....	363
7.13.8.7	Syuds.Pivot .....	365
	Report Design .....	367
	Sample Output .....	370
7.13.8.8	Repeat .....	372
7.13.8.9	Syuds.Store .....	373
7.13.8.10	Sum .....	374
7.13.8.11	TableAlias .....	376
7.13.8.12	Syuds.TimeSlice() .....	377
7.13.8.13	Top .....	379
7.13.8.14	Union .....	380

<b>8 Workflow Overview</b>	<b>382</b>
8.1 Business Benefits	382
8.2 The COINS Workflow toolkit	383
8.3 Example Workflow	384
8.3.1	384
8.4 Workflow Groups	385
8.5 Workflow Roles	386
8.5.1 Example Roles	387
8.6 Workflow Responsibilities	388
8.6.1 Example Responsibilities	388
8.7 User Workflow Responsibilities	390
8.8 Workflow Delegation	392
8.8.1 Self-Service Workflow Delegation	394
8.9 Workflow Templates	395
8.9.1 WF Template - Details Tab	396
8.9.2 WF Template - Stages Tab	399
8.9.2.1 WF Stages - Appointment	402
8.9.2.2 WF Stages - Delay	405
8.9.2.3 WF Stages - Email	407
8.9.2.4 WF Stages - Fork	409
8.9.2.5 WF Stages - Merge	411
8.9.2.6 WF Stages - Process	413
8.9.2.7 WF Stages - SMS	415
8.9.2.8 WF Stages - Stage	417
8.9.2.9 WF Stages - Task	423

8.9.3 WF Template - Diagram Tab .....	426
8.9.4 WF Template - Check Tab .....	435
8.9.5 WF Template - Paths Tab .....	436
8.9.6 WF Template - Active Tab .....	437
8.9.7 WF Template - Complete Tab .....	438
8.9.8 WF Template - Outstanding Tab .....	439
8.10 Workflow Deployment to Companies .....	441
8.11 Workflow Monitor .....	443
8.11.1 Workflow Monitor Workbench .....	445
8.11.1.1 WF Monitor - Monitor Tab .....	446
Workflow Monitor diagram .....	446
8.11.1.2 WF Monitor - Active Tab .....	448
8.11.1.3 WF Monitor - Completed Tab .....	449
Completed Tab .....	449
8.12 Workflow Summary .....	450
8.12.1 WF Summary - Tasks Tab .....	451
8.12.2 WF Summary - Stages Tab .....	452
8.12.3 WF Summary - History Tab .....	453
8.12.4 WF Summary - Variables Tab .....	454
8.13 Workflow User Variables .....	455
8.14 Workflow Parameters (System Parameters) .....	456
8.15 Database Triggers .....	457
8.16 14 Import/Export Workflow Template .....	460
8.17 15 Launching Workflow .....	461
8.18 Using a Choose Action .....	462



8.1917 Method to force the closure of Workflow: .....	463
<b>9 Introduction or Executive Summary .....</b>	<b>464</b>
9.1 soapUI .....	465
9.2 Licencing .....	466
9.2.1 Additional Licencing .....	466
9.2.2 Roles .....	469
9.2.3 User Maintenance .....	471
9.3 Viewing Installed Services .....	472
9.4 Testing a Service Connection (Using soapUI) .....	476
9.5 ESB COINS User .....	487
9.6 User Defined Services .....	489
9.6.1 Datasets .....	490
9.6.1.1 Function .....	490
9.6.1.2 Dataset .....	490
9.6.1.3 Page .....	492
9.6.1.4 Adding the Function to the Web services Menu .....	493
9.6.1.5 Testing the Service .....	499
9.6.2 Pages .....	503
9.6.2.1 Function .....	503
9.6.2.2 Page .....	504
Body Form .....	506
Update Form .....	506
Detail Form .....	507
9.6.2.3 Adding the Function to the Web services Menu .....	508
9.6.2.4 Testing the Service .....	513

Using the ID field when adding multiple records .....	516
9.6.3 Calculation Programs .....	518
9.6.3.1 Defining the Calculation .....	518
9.6.3.2 Testing Calculation Programs (run, run\$) .....	520
9.6.3.3 Using Calculation Programs with Web Services .....	523
Create the Function .....	523
Create the Page .....	524
Adding the Function to the Web services Menu .....	526
9.6.3.4 Testing the Service .....	531
9.6.4 Get/Set .....	534
<b>10 Building COINS Desktops - Overview .....</b>	<b>540</b>
10.1 Desktop Structure .....	541
10.1.1 Creating the Functions (Manually) .....	543
10.1.1.1 Desktop Functions .....	543
10.1.1.2 Desktop Tab Functions .....	543
10.1.2 Add the Desktop to a user profile .....	544
10.1.3 Desktop Section Functions .....	547
10.1.4 Uploading Desktop Images .....	550
10.1.5 Tile Rows and Columns .....	552
10.1.5.1 Desktop – Tile Rows .....	553
10.1.5.2 Desktop – Tile Columns .....	554
10.1.6 Add the Desktop to a user profile .....	555
10.1.7 Alternative Desktops .....	558
10.1.7.1 User Maintenance .....	558
10.1.8 Desktop Import/Export .....	560





10.1.8.1	User Maintenance .....	560
10.1.8.2	Export .....	560
10.1.8.3	Import .....	563
10.1.9	Smart Tiles .....	565
10.1.10	Info Tiles .....	568
10.1.11	Using Calculation Programs to prepare data for the Desktop .....	572
10.1.12	Desktop Tile Values .....	575
10.1.12.1	Examples of Desktop Tile Values .....	575
10.1.13	Tile Options .....	579
10.1.13.1	NamedFilter Example .....	581
Desktop	Tile Options .....	581
Desktop	.....	582
10.1.13.2	InitContainer Example .....	585
Desktop	Tile Options .....	585
Desktop	.....	585
10.1.14	Tables on Tiles .....	587
10.1.14.1	Tables : Example 1 - Direct Query against the database ....	588
Create the Function	.....	588
Create the Page	.....	588
Desktop	.....	589
10.1.14.2	Example 2 - Using a Data Set .....	590
Create the Function	.....	590
Create the Dataset	.....	590
Desktop	.....	592
10.1.15	Charts on Tiles .....	593



10.1.15.1Dataset driven Charts .....	593
Create the Function .....	593
Create the Dataset .....	593
Create the Page .....	595
Create the Page .....	596
Desktop .....	600

## 2 Database Overview

A database is a collection of records stored in a computer in a systematic way, such that a computer program can consult it to answer questions. For better retrieval and sorting, each record is usually organized as a set of data elements (facts). The items retrieved in answer to queries become information that can be used to make decisions. The computer program used to manage and query a database is known as a database management system (DBMS). The properties and design of database systems are included in the study of information science.

The central concept of a database is that of a collection of records, or pieces of knowledge. Typically, for a given database, there is a structural description of the type of facts held in that database: this description is known as a schema. (You can view the coins schema using the database enquiry screen in coins OA – see accompanying documentation).

The schema describes the objects that are represented in the database, and the relationships among them. There are a number of different ways of organizing a schema, that is, of modelling the database structure: these are known as database models (or data models).

## 2.1 Relational Database Model

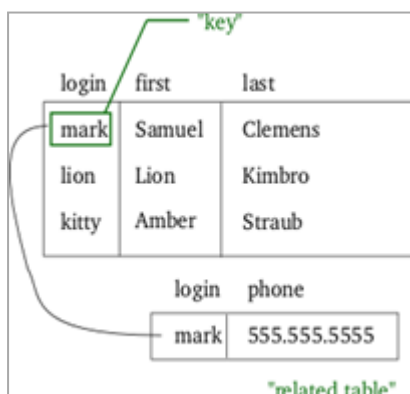
The model in most common use today is the relational model, which represents all information in the form of multiple related tables each consisting of rows and columns.

A relational database is a database based on the relational model. Strictly speaking the term refers to a specific collection of data but it is invariably employed together with the software used to manage that collection of data. That software is more correctly called a relational database management system, or RDBMS.

An important feature of relational systems is that a single database can be spread across several tables. This differs from flat-file databases, in which each database is self-contained in a single table.

On Relational Database Model represents relationships by the use of values common to more than one table

In the relational model some bit of information was used as a "key", uniquely defining a particular record. When information was being collected about a user, information stored in the optional (or related) tables would be found by searching for this key. For instance, if the login name of a user is unique, addresses and phone numbers for that user would be recorded with the login name as its key.



## 2.2 Tables and Modules

The table structure of the coins database has been designed to be organized with a direct relation to the business processes and modules of the system. One of the main attributes of this design is the naming convention used on the schema.

Coins have aimed to use a naming convention that would make it easy to identify which tables are used by which module. On most cases the first two letters of the table will point to the module code of the system.

Examples:

ap\_ Purchase Ledger  
ar\_ Sales Ledger  
cb\_ Cash Book  
ci\_ Central Repository

The above is the standard convention but there are tables in the database which do not conform to this convention of which users should be aware – main examples are payroll and all system information which either do not use underscores but use hyphens or do not break the table names. The Database Enquiry contains all the information as required

The table name will also contain a descriptive element, for example ap\_invoice is the Purchase Ledger invoice table and ap\_invdist the table which contains its associated distribution records.

Each table has a three letter ID, this is used as reference throughout coins – and is often used as the prefix of a field name. In the example of ap\_invoice this ID is ain, therefore the field name for the Purchase Ledger Invoice balance is ain\_balance.

To reference a field the syntax is:

`{tablename}.{fieldname}`

For example :

ap\_invoice.ain\_balance

## 2.2.1 Database Structure

The coins database is based on various levels, the top level being the Central Repository. Information held in the Central Repository is not COINS Company specific and is available across the system.

The main pieces of information held in the Central Repository are :

**CI** Company Information

**PI** Project Information

**TI** Technical Information

**PP** People Information

In addition to the Central Repository, system information such as Users, Functions, Printers are also held at this top level.

**SY** System

**MS** Menus and Functions

**PM** Print Manager

**XL** Translations and Language

**IB** Insurances and Bonds

**MK** Marketing

Most data in the Coins database is actually held at COINS Company level. Even though only one company may be used the company details will need to be used to access the data. In each instance the company number is held on each table in the kco (current logged in company) field.

There are a set of tables which relate directly to company information – configuration table etc, in addition to generic tables such as Batches – these are held in the co module.

**CO** Company

## **2.2.2 Company Specific Modules**

**GL** General Ledger  
**JC** Contract Status Ledger (Job Costing)  
**CB** Cash Book  
**AP** Purchase Ledger (Accounts Payable)  
**AR** Sales Ledger (Accounts Receivable)  
**SC** SubContract Ledger

## **2.2.3 Process Specific Modules :**

**CS** Contract Sales

**FM** Facilities Management  
**SM** Valuations (Site Manager)  
**SW** Small Works

### **House Builders Modules**

**BQ** Bill of Quantities  
**HS** House Sales  
**LA** Land Appraisal  
**VP** Valuations and Payments  
**WF** Workflow

### **Payroll and HR**

**CR** Credit Control  
**HR** Human Resources  
**EX** Expenses  
**PR** Payroll

### **Plant, Assets and Stock**

**CM** Components  
**FA** Fixed Assets  
**FL** Fleet  
**PC** Plant Control  
**SO** Sales Orders  
**ST** Stock



## **Procurement**

**PO** Procurement

Other Modules :

**BP** Professional Billing

**CR** Credit Control

**DC/DM** Document Control / Management

**IB** Insurance and Bonds

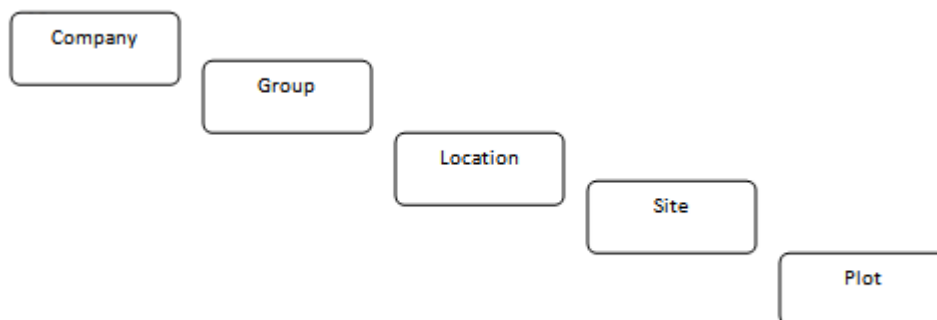


## 2.3 Summary and Detail Tables

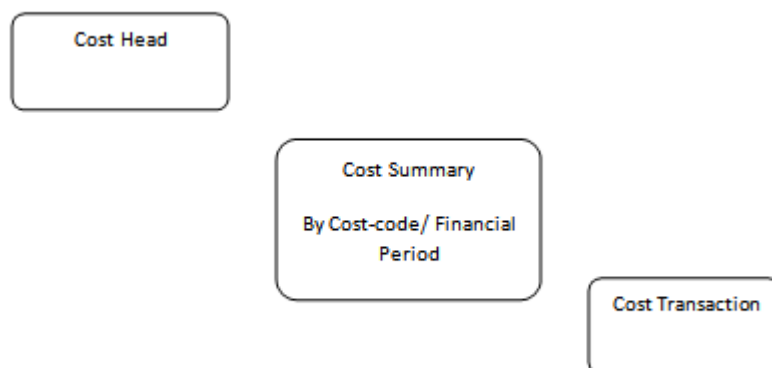
Coins has designed its database to match the processes of the industry and therefore the tables mirror the process components. Each of the modules will cover different process and within each module we can find different sub-processes. The organizational structure of the company is also related to the structure of the tables.

Each of these sub-processes are also divided in tables that will hold information down to the lowest level of detail and there will also be tables that will summarize that information at different levels (depending to the process these could be dates, codes etc).

Company Organizational Structure & Contract Structure Example (House Builders):



Cost Transactions Structure Example (House Builders):



## 2.4 Open Items

In addition to the Summary and detail information, COINS also has specific tables for open transactions in the database. This is to enhance performance when maintain and reporting on current data.

For example every PL Invoice that has not been paid, or has been part paid will have an associate record in the PL Invoice Open Item table. Once an invoice has been fully paid the open item record is deleted.

It is therefore recommended that when enquiring or reporting on open items that it is the open item record which is used as the basis of the query.

For example :

```
FOR EACH ap_invopen WHERE ap_invopen.kco = {kco},  
EACH ap_invoice OF ap_invopen
```

Each of the tables which contain transactional data will have an associate open item table.

## 2.5 Indexes

Databases can take advantage of indexing to increase their speed (Dataset retrieval using queries). The most common kind of index is a sorted list of the contents of some particular table column, with pointers to the row associated with the value. An index allows a set of table rows matching some criterion to be located quickly.

The order that columns are listed in the index definition is important. It is possible to retrieve a set of row identifiers using only the first indexed columns. However, it is not possible or efficient (on most databases) to retrieve the set of row identifiers using only the second or greater indexed column.

For example, imagine a phone book that is organized by city first, then by last name, and then by first name. If given the city, you can easily extract the list of all phone numbers for that city. However, in this phone book it would be very tedious to find all the phone numbers for a given last name. You would have to look within each city's section for the entries with that last name.

Each coins table has one or more Indices. An Index is built up of several fields in a record which in combination will assist the query in narrowing down the number of records which will be read to determine which meet the query requirements.

Each Table has a Primary (the index used by default unless you determine otherwise in your query) and a Unique key (the combination of these fields in a single record is always unique).

However coins OA will use the most appropriate index for your query.

An example of how an index would work is to use the Current Logged in Company (kco) in addition to Contract Number (job\_num) to search for a particular Contract. Another would be to use Current Logged in Company (kco) plus Order Type (tip\_type) where you would query only where tip\_type = "TRADE", the query would immediately know only to search through Subcontract Orders to find orders which matched the other criteria rather than search every single order.

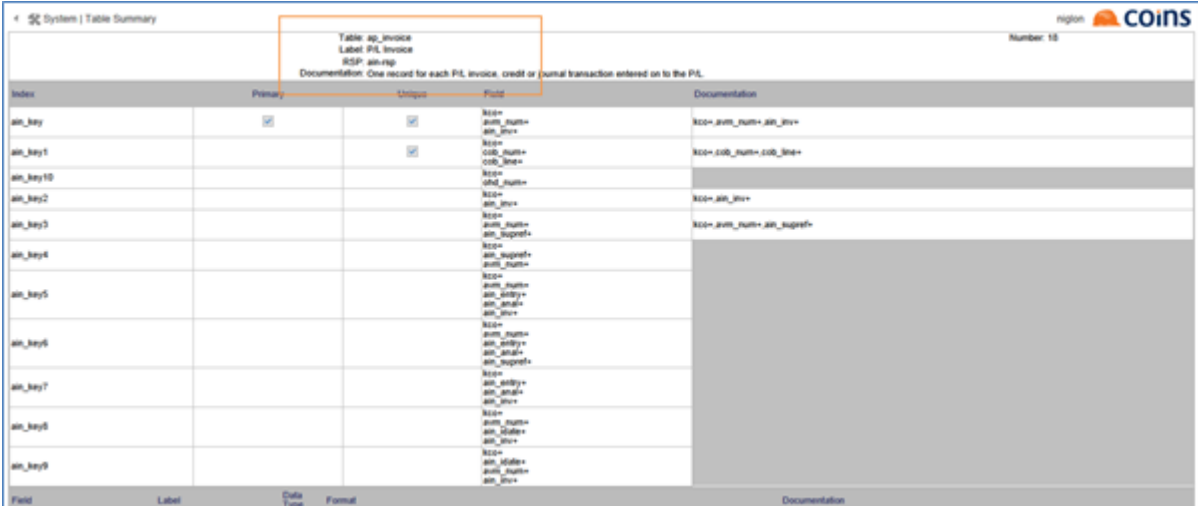
## 2.6 Record Service Procedures

Every table in the coins database has a RSP (Record Service Procedure). Each of these RSP's provides the OA Reporting tool with the business logic required to extract the appropriate data from the database.

In the Database Enquiry you can see the RSP under the Table Code and its Label. The RSP's have a naming convention –

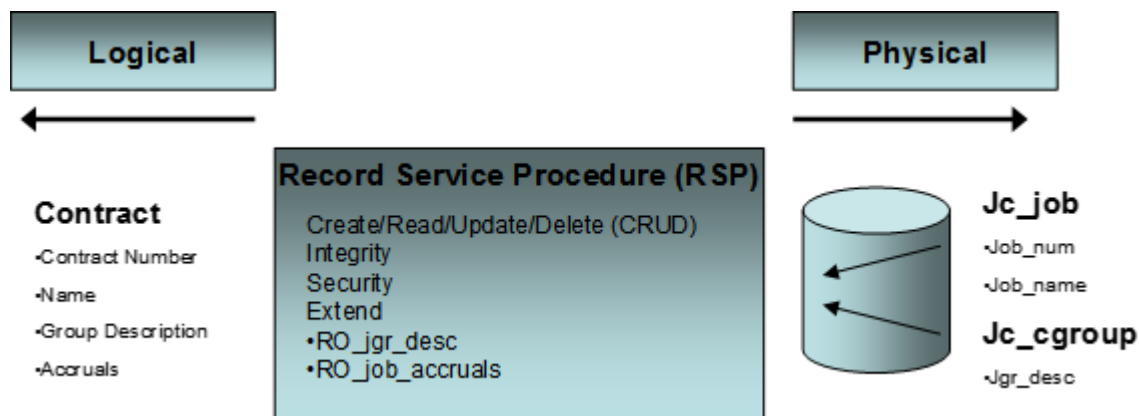
{table-ID}-rsp.p.

Where the Table ID is as shown in the Database Enquiry (you may also hear this referred to as the Table Acronym or TLA).



Index	Primary	Unique	Null	Documentation
pk1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	NO	NO= inv_num+ inv_inv+
pk2	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	NO	NO= inv_num+ inv_inv+
pk3	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	NO	NO= inv_num+ inv_inv+
pk4	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	NO	NO= inv_num+ inv_inv+
pk5	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	NO	NO= inv_num+ inv_inv+
pk6	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	NO	NO= inv_num+ inv_inv+
pk7	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	NO	NO= inv_num+ inv_inv+
pk8	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	NO	NO= inv_num+ inv_inv+
pk9	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	NO	NO= inv_num+ inv_inv+
pk10	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	NO	NO= inv_num+ inv_inv+

RSP's control, amongst other things, the basic table update functions for that table. Each RSP has a common set of methods that define standard behaviour for the object. They control record scope and locking, security, default values on creation, data integrity rules etc.

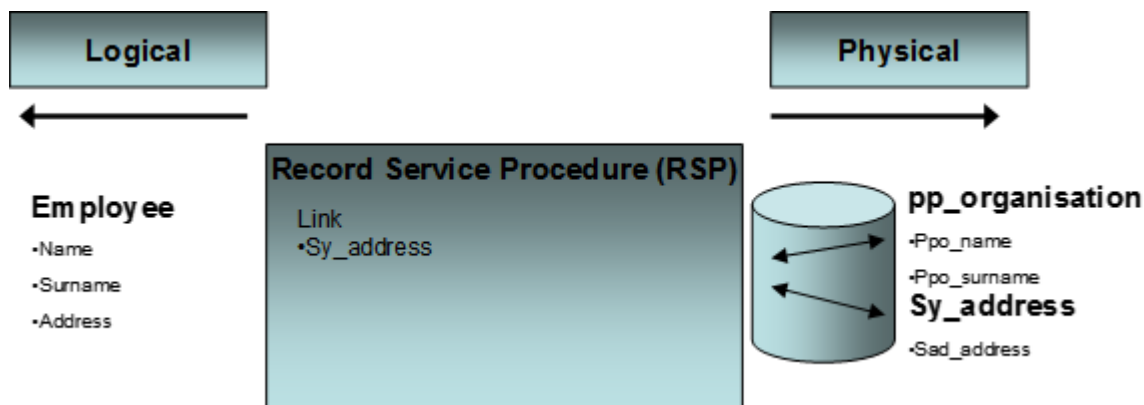


The RSP can also extend the database table to produce a logical view of data for the business logic to work with. A by-product of this is that we are to provide an XML field on every record in the database in which the user can configure their own extended fields for use in the presentation layer. This enables client-specific fields to be added to screens and included on reports. The RSP handles the translation of the data from the logical record buffer to the physical fields in the physical database.

The RSP is also able to de-normalise the database for the purposes of logical data access. For example, the contract record (jc\_job) in the database is linked to many other tables including the contract group table (jc\_cgroup). To show the group description on a screen or report it might be expected that the interface designer would have to build a query to link the contract record with the appropriate group record and then display the group description from that record. The RSP extends (for read-only purposes) the table and de-normalises the data and makes the contract group description available as just another field on the contract table. The data remains physically in the jc\_cgroup table in the database, but to the business logic and presentation layers it is shown in a more logical place which is on the contract record.

The same principle is applied to more complex calculation fields. For example, the value of accruals on a contract is a highly complex calculation involving many tables (purchase orders, order lines, goods received notes, etc). Again, this data is made available in the logic buffer in the RSP so that to the interface designer it is just another field on the contract record. They are simply able to paste, say, the contract number, contract name, group description, and accrual value on to a page without any need to know where each of the bits of data is coming from.

The RSP is also able to simplify the database for updates.



There are instances where common database tables are reused in many areas of the application. An example is the address table sy\_address. This holds the address details for an employee record in the HR system. In this instance the RSP is able to link the two records together, presenting a single logical table to the business logic and presentation layers. The updates are performed on this logical buffer in the RSP and it is only the RSP that knows that the data is split into two separate records when written back to the database.

The RSP also has an audit layer so that as the logical record is committed back to the database, changes on the logical table can be recorded in the audit records. Auditing can be performed as part of the managed data source through the use of triggers, but in this instance the audit records created are an audit of the physical data and it is much more difficult to reconstruct the separate physical table and field audit records in to a logical view of the record at a later stage.

The RSP controls all data access through to the database. COINS can insert bespoke trigger code in that RSPs that can act on data as it is committed back to the database. For example, this can be used to push changes in business data in COINS out to a data warehouse application by creating XML messages from COINS that are then consumed by a data warehouse load interface. In this way it is possible to keep a data warehouse up to date with live business data.

## 2.7 Read Only Fields

In addition to the standard tables and fields in the coins database, Open Architecture also uses the RSP's to provide access to certain calculated and non standard fields. These are known as "RO" or Read Only fields and are also fully documented in the Database Enquiry.

Field Name	Description	Format	Restrictions	Notes
RO_Inv_RoutingCode	International Bank Routing Code	character X(5)	RO	
RO_Inv_Scib_Details	Supplier Tax Details	character X(80)	RO	If the supplier provides subcontract labour (Inv_Scib=TRUE) then this is a complete description of the CIS details for the subcontractor.
RO_Inv_Tile_Age	Ageing Tile Age	character X(10)	RO	Flag "Age"
RO_Inv_Tile_Amt	Ageing Tile Amount	character X(10)	RO	Flag "Amount"
RO_Inv_Tile_Key	Ageing Tile Key	character X(10)	RO	Flag "Key"
RO_Inv_Tile_QryAmt	Ageing Tile Query Amount	character X(12)	RO	Flag "Query Amount"
RO_Inv_Total	Outstanding Total	decimal -9,999,999,999.99	RO	Total outstanding invoice/payments for the supplier. Sum of the open invoices and payments in account currency.
RO_Inv_Uncleared	Uncleared to Pay	decimal -9,999,999,999.99	RO	The total of invoices that are due to be paid and are held.
RO_Inv_WFVar	Active Workflow Variable Value	character X(30)	RO	
RO_BalD[2]	Discount Lost	decimal -9,999,999,999.99	RO	Discount lost in the year taken from ap_venbal for the current GL year. Array 1=current GL year, 2=next year.
RO_BalT[2]	Discount Taken	decimal -9,999,999,999.99	RO	Discount taken in the year taken from ap_venbal for the current GL year. Array 1=current GL year, 2=next year.
RO_BalPay[2]	Payments	decimal -9,999,999,999.99	RO	Payments made in the year taken from ap_venbal for the current GL year. Array 1=current GL year, 2=next year.
RO_BalPur[2]	Purchases	decimal -9,999,999,999.99	RO	Purchases (Invoices) in the year taken from ap_venbal for the current GL year. Array 1=current GL year, 2=next year.
RO_BalFin[2]	Financial Year	character X(4)	RO	Current GL year. Array 1=current GL year, 2=next year.
RO_Inv_Store	Company Data Store	decimal -9,999,999,999.99	RO	Standard store options. See co_config RW_Inv_Store
RO_ContractAccruals_Amt	Contract Accruals by Contract	decimal -9,999,999,999.99	RO	The contract accruals for the supplier. The contract for which accruals are required. -CAN-DO list of cost heads to exclude.
RO_ContractCosts_Amt	Contract Costs by Contract	decimal -9,999,999,999.99	RO	The contract costs for the supplier. The contract for which accruals are required. -CAN-DO list of cost heads to exclude.
RO_Inv_Currency	Reporting Currency	character X(4)	RO	If RSP_base is set to YES then the company base currency otherwise the supplier account currency. Whether notes have been entered against this company.
RO_Flag_Notes	File for Red/Clear Flag or no Flag picture	character X(80)	RO	If at least one note is marked by a Red Flag, the flag will appear here as a Red Flag. To see the notes, click the flag icon. See Comments Notes.
RO_Inv_1	Turnover Q1	decimal -9,999,999,999.99	RO	Turnover in Q1 for the year held in variable RS_Inv_Year
RO_Inv_2	Turnover Q2	decimal -9,999,999,999.99	RO	Turnover in Q2 for the year held in variable RS_Inv_Year
RO_Inv_3	Turnover Q3	decimal -9,999,999,999.99	RO	Turnover in Q3 for the year held in variable RS_Inv_Year
RO_Inv_4	Turnover Q4	decimal -9,999,999,999.99	RO	Turnover in Q4 for the year held in variable RS_Inv_Year
RO_Inv_Changed	Record Changed	character X(24)	RO	The date and time of the most recent change to the record, and the user ID of the user who changed it. The last changed date, time and user id.
RO_Inv_Created	Record Created	character X(17)	RO	The date and time that the record was created. The created date and time.
RO_Inv_Code	Subcontractor Account	character X(10)	RO	If Inv_Scib = TRUE and the subcontract (sc_main) record exists for this supplier then this shows the subcontractor account code otherwise blank.
RO_Supplier	Group Supplier	character X(3)	RO	The company the plant item is tied from. If variable RS_Group = YES then if the group account (Inv_Group) is non blank then the group account otherwise the supplier account code.
RO_Inv_Branch	Bank Branch (Workflow)	character X(30)	RO	
RO_Inv_Bank_Name	International Bank Name (Workflow)	character X(30)	RO	
RW_Inv_Notes	Notes	character	RW	Any notes pertaining to the supplier.

Although these fields have certain restrictions, they are incredibly powerful when used in enquiries and reports.

In most instances RO\_ fields will provide information from related tables to the main queried table – for example summary cost information at Contract Level, or descriptions from an associated Lookup Table without the Page or Report designer having to query and access many tables from the coins database.

Many of the calculated fields reflect similar fields to the coins + Configurable Reporter, such as Accruals, Costs and Revenue fields. These fields can then be passed parameters to enhance the information returned to a report. Typically these fields can be limited by dates, values and financial periods as well simply parameters such as "TD" for a To Date value.

In the Database Enquiry RO\_ fields are shown in a format as the example below. Any parameters immediately after the caret are mandatory; each parameter is then separated by a pipe. Any parameters which are encapsulated in square brackets are optional.

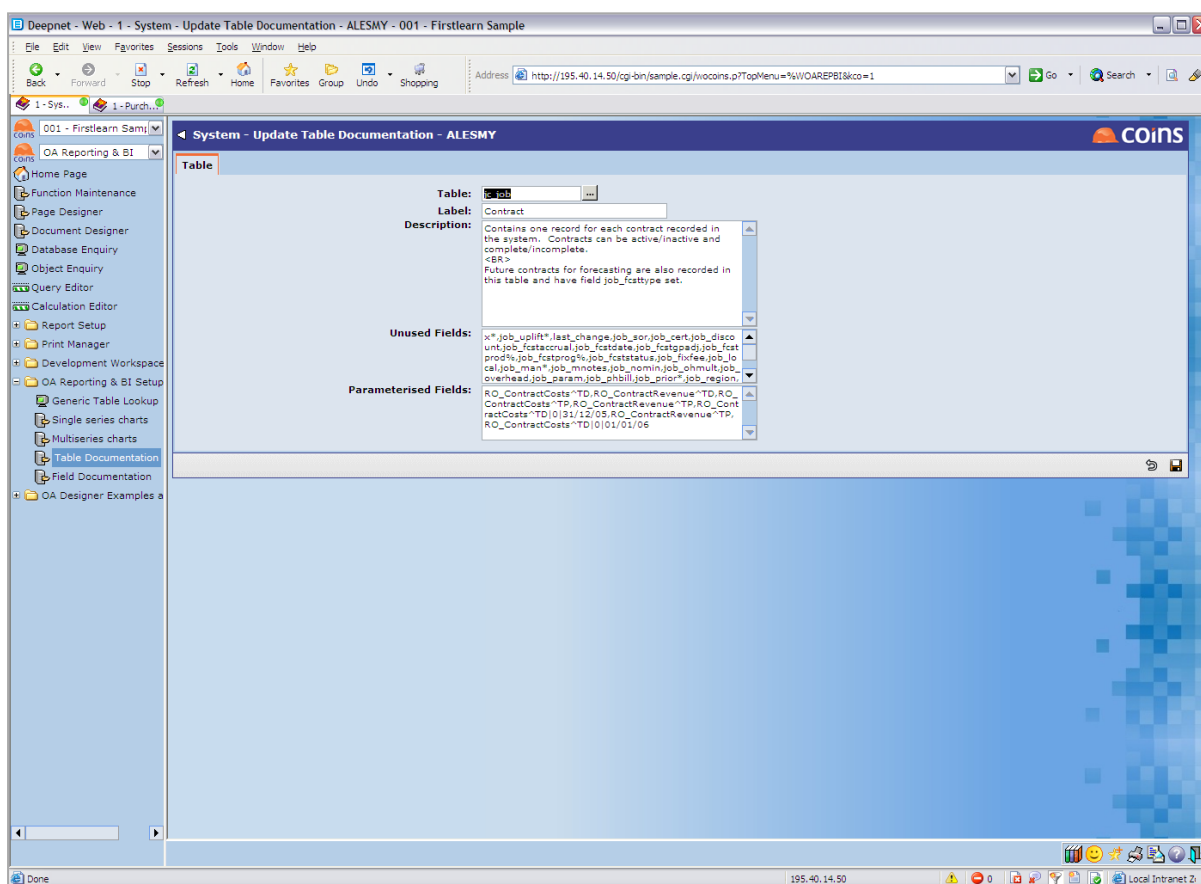
```
RO_ContractCosts^<PeriodType>[<PeriodOffset>[<FDate>[PhaseMasks[CostcodeMasks  
[CategoryMasks[AnalysisMask]]]]]]
```



## 2.7.1 Exposing RO Fields

The benefit of using this method is the fields will be selectable by the user as report columns when building the report, however as RO Fields they will NOT be available for sorting and grouping. The disadvantage is that the fields will be hard-coded to the state they are exposed as and any change to their functionality must be made by the administrator and cannot be changed by the report writer users. For this reason, it is recommended that RO fields are used in calculations within the Report rather than by Exposing.

Go to the OA Reporting and BI Setup and Table Documentation and select the table in which the field you wish to pass parameters to.



The Parametrised Fields section then needs to be populated with the appropriate RO\_ field and any parameters that are required. (The parameter information can be found in the Database Enquiry.

A single field can be documented more than once with different parameters, for each a balance field can be documented with several dates, such as a year of period end dates.

Examples of an RO\_ field and associated parameters are :


**jc\_job.RO\_ContractCosts^TP|-3** (This Period with period offset -3)

**cs\_certificate.RO\_cst\_ctd\_cum^10** (Contract Sales Certificate Item – Cumulative to Date Certificate Line 10)

Once a RO\_ field has been entered in the Table Documentation with appropriate parameters it will then become available in Report Writer for selection as per standard database fields.

It will also appear in the Database Enquiry with a field type of EX

## 2.8 Create and Maintain Lookups

Lookups apply to normal (fill in) fields, and to 'disabled with lookup' fields. Fields for which a lookup has been set up have a lookup button  next to them; disabled with lookup fields have lookup and clear buttons next to them:

The lookup button launches a separate browser, from which the user can select a record. The lookup then returns values to populate one or more fields on the form. If a user adds a new record whilst in a lookup (assuming this is allowed) then when they save the new record it will automatically be selected and returned from the lookup.

In Lookup Maintenance, enter a record for the field to provide a lookup for.

In the Lookup Function field enter the function to call for lookup. This function is a normal browse/list type page (and could even be say the maintenance function - the maintenance buttons will be retained if access security permits). Links are removed.

To provide different lookups for the same field on different functions, enter separate records with the function code in the Function field. If the Function field is blank, COINS uses this as the default lookup for the field.

The inline lookups are built using the existing page for the lookup button but there are two new options you can specify in the parameters of the lookup. quickFields and quickShow.

By default the inline lookup uses the same field (keycode/lookupcode) that is used by the lookup page and does a begins on that field. If you need to modify the field to another one or multiple then you can do this with quickFields. This is a comma separated list of field names to be matched when typing in the characters in the fill in. Example is poh\_attention where quickFields is set to ppc\_name,ppc\_surname so the matching is on either of those fields.

In the parameter field for the lookup, it is possible to specify the following:

- keyfield=field. By default (if you do not specify a keyfield), the lookup will return the value of the field from the lookup record that has the same name as the field you are looking up (that is, the one specified in the Field field in Lookup Maintenance). By specifying a keyfield, you can get the value of a different field. For example, the Field may be avm\_factnum, but the keyfield is specified as avm\_num.
- lookupCode=field-list. A list of other fields on the form that you want to populate (with the corresponding values from the keyCode list).

- `keyCode=field-list`. A list of other fields in the same table that you want to get values from. If `keyCode` is blank, the values are taken from the field specified in `lookupCode`.
- `contextFields=field-list`. A list of fields on the current page that are required for the context of the lookup (for example, the supplier code if you are looking up invoices). These will be passed to the lookup program and used in the query. See `RO_hrs_desc` lookup, which requires the current value of `hrg_code` from the form to be able to show the appropriate sub groups.
- `quickFields=field-list`. A list of fields which the inline lookups will query (For example: `quickFields=job_num,job_name` - will return the following lookup when entering characters in the lookup field. It currently uses a begins to provide the search. Example will give you any `job_num` beginning with 10 or any `job_name` beginning with 10 as a drop down list. A max of 10 entries will be shown and if further information is available on the 11th line of the inline you will see ... If there is an exact match to the entry then the field will be filled in on leaving the field via tab or enter. By partial input the lookup button will reflect this search filter, by not entering any input the lookup filter will be remembered from previous entry.
- `quickShow=field-list`. This will provide the fields to be displayed in the inline lookup. These fields can be either database fields or RO fields. It is advisable to keep this list to a minimal number.
- `quickSort=field+`. This will predetermine the order the inline information is displayed.

### 3 OA Query Language

COINS OA uses a simplified version of the Progress 4GL query language in combination with RSP's (Record Service Procedures) to extract the data for reports and enquiries (for further information on RSP's- see later in this guide).

COINS OA uses the query to decide which records are accessed from the coins database from the database. In response to a query, the database returns a result set, which is just a list of rows containing the answers. The Page/Report Design will determine which fields from these records are displayed (either on screen or in a report).

The simplest query is just to return all the rows from a table, but more often, the rows are filtered in some way to return just the answer wanted.

The flexibility of relational databases allows programmers to write queries that were not anticipated by the database designers. As a result, relational databases can be used by multiple applications in ways the original designers did not foresee, which is especially important for databases that might be used for decades. This has made the idea and implementation of relational databases very popular with businesses.

## 3.1 FOR EACH

To begin a query in OA, the first statement must begin FOR EACH followed by a table name.

Example Query on the coins database to retrieve all contracts (jc\_job)

FOR EACH jc\_job

1. jc\_job is the name of the table in the COINS database
2. The FOR EACH statement starts a block of code that iterates once for each contract record (hence the syntax FOR EACH)

## 3.2 WHERE

Simply specifying the table with a FOR EACH statement in a query is okay, assuming we want every record from the selected table, but in practice we would normally want to restrict the number of records returned in some way. In COINS, transactional data is held at company level.

Even though you may only have one company in your organisation, the data is still recorded with a company identifier. COINS uses the field kco to identify the company number.

Most queries will need to specify the kco values to ensure that the records returned relate specifically to the company you are reporting on.

The WHERE statement is used to add a constraint to the query and may refer to a constant, filed name, variable name or expression whose value you want to use to select records

Example Query on the coins database to retrieve all contracts (jc\_job) that belong to company 1:

```
FOR EACH jc_job WHERE kco = 1
```

In the example above we have used '=' as the comparison operator. There are a number of others than may be used with the WHERE statement. These are listed in the table below:

Keyword	Symbol	Description
EQ	=	Equal to
NE	<>	Not equal to
GT	>	Greater than
LT	<	Less than
GE	>=	Greater than or equal to
LE	<=	Less than or equal to
BEGINS	Not applicable	A character value that begins with this substring.
MATCHES	Not applicable	A character value that matches this substring, which can include wild card characters
		The expression you use to the right of the MATCHES keyword can contain the wild card characters:

Keyword	Symbol	Description
CONTAINS	Not applicable	<p>An asterisk (*) represents one or more missing characters.</p> <p>A period (.) represents exactly one missing character.</p> <p>A database text field that has a special kind of index called a WORD-INDEX</p> <p>The WORD-INDEX indexes all the words in a field's text strings, for all the records of the table, allowing you to locate individual words or associated words in the database records, much as you do when you use an Internet search engine to locate text in documents on the web..</p>

The WHERE statement can be followed by any expression that identifies a subset of the data using AND/OR to join multiple tests.

Example Query on the coins database to retrieve a specific contract (field job\_num) for Company 1 from table jc\_job

```
FOR EACH jc_job WHERE jc_job.kco = 1
AND jc_job.job_num = '123456'
```



## 3.3 Joining Tables

Often, data from multiple tables gets combined into one, by doing a join. Conceptually, this is done by taking all possible combinations of rows (the "cross-product"), and then filtering out everything except the answer.

To begin each join a comma should end the previous statement before beginning the next one. DO NOT add a comma to the end of the last statement as this will result in an error.

### 3.3.1 EACH

FOR is only used for the first table in the query, all subsequent tables must be accessed with EACH to start an iterating query that will find a single record on each pass

To establish a join, the table(s) you are adding to the query must have some relation to one or more tables already in the query.

Example Query on the coins database to retrieve all costheads (jc\_costcode) that belong to contracts (jc\_job) that belong to the logged in Company

```
FOR EACH jc_job WHERE jc_job.kco = {kco},  
EACH jc_costcode WHERE jc_costcode.kco = jc_job.kco  
AND jc_costcode.job_num = jc_job.job_num
```

If you do not use the EACH keyword for a subsequent table then you must use one of the following to obtain a single record:

### 3.3.2 FIRST

Uses the criteria in the record-phrase to find the first record in the table that meets that criterion.

Progress finds the first record before any sorting.

### 3.3.3 LAST

Uses the criteria in the record-phrase to find the last record in the table that meets that criterion.

Progress finds the last record before sorting.

The FIRST and LAST keywords are especially useful when you are sorting records in a table in which you want to display information. Often, several related records exist in a related table, but you only want to display the first or last related record from that table in the sort. You can use FIRST or LAST in these cases.

### 3.3.4 OF

Some of the tables in the COINS database share a relationship based on common field names between record and table that also participate in a UNIQUE index for either record or table. All OF relationships within the coins database are detailed in the database enquiry and appear for each table in the form similar to:

From	To	Join To	Documentation	Code
1	*	ap_invsat	ap_invsat OF jc_job	ap_invsat.kco=jc_job.kco AND ap_invsat.job_num=jc_job.job_num
*	1	ar_custum	Links to the Customer Summary record for the customer of this Contract.	ar_custum.kco=jc_job.kco AND ar_custum.rcm_num=jc_job.rcm_num

Where such a relationship exists, the OF statement may be used to relate one table to another. So in our earlier example we used the query:

```
FOR EACH jc_job WHERE jc_job.kco = {kco},
EACH jc_costcode WHERE jc_costcode.kco = jc_job.kco
AND jc_costcode.job_num = jc_job.job_num
```

An OF relationship exists between jc\_job and jc\_costcode as can be seen in the database enquiry for jc\_job:

1	*	jc_costcode	jc_costcode OF jc_job	jc_costcode.kco=jc_job.kco AND jc_costcode.job_num=jc_job.job_num
---	---	-------------	-----------------------	--

So we can re-write this query as:

```
FOR EACH jc_job WHERE jc_job.kco = {kco},
EACH jc_costcode OF jc_job
```

## 3.4 Curly Braces

The functionality of {}'s is to specify a place holder in fields and calculations into which a value can be passed.. When using {}'s around a field the use of quotes is required if the field is a character field. The use of double or single quotes is acceptable.

The only thing to be aware of is that when using '{field}' replacement on a character field is that if the information within the field could contain an apostrophe (for example- J O'Connor) then the apostrophe would cause close to the single quote and you will get a symbol not found(Connor) Error. To overcome this error the use of double quotes "{field}" is the answer.

The use of {}'s in calculations is possible on all field values **except within the DataSets and the calculate conditions on a report**. In these instances it is necessary to always qualify out the field with the table name.

```
{RO_ContractCosts^TD|0|{RS_glp_fdate__2}}  
would be written as:  
jc_job.RO_ContractCosts^TD|0|{RS_glp_fdate__2}.
```

The use of the table name is allowed in all calculations but whereas in most instances the formatting of the result is suppressed, within the calculate condition it is not and therefore the comma in a result of a figure in excess of 1,000 may result in an error in syntax in a calculation. (NB. Please note that the replacement on parameters of an RO field is still acceptable).

Within the OA reporter/screens we use curly braces {} as a method to pass values to a query or a report or a page. Enclosed within the curly braces you specify the commands, RS\_fields, or other data you need to communicate across or within objects. {kco} is a common usage, and is used to place the current logged in company number into the query.

The next example gets information from jc\_job and inherits the Company Number from the system, retrieving the company number the user is logged into.

```
FOR EACH jc_job WHERE jc_job.kco = {kco}
```

## 3.5 Outer-Join

Specifies a left outer join between record and the table (or join) specified by the previous Record phrase(s) of an OPEN QUERY statement. A left outer join combines and returns data from the specified tables in two ways. First, the records selected for the table (or join) on the left side combine with each record selected using the OF or WHERE options from the table on the right (record). Second, the records selected for the table (or join) on the left side combine with unknown values (?) for the fields from the table on the right (record) for which no records are selected using the OF or WHERE options. The join is ordered according to the given sort criteria starting with the left-most table in the query.

```
FOR EACH co_batref WHERE co_batref.kco = {kco}
AND co_batref.cbt_type = 'CSCERT'
AND co_batref.cob_num = {cob_num},
EACH cs_certificate WHERE cs_certificate.kco = {kco}
AND cs_certificate.cst_ref = co_batref.cbr_txn,
EACH cs_certdist OF cs_certificate
FIRST gl_acct OF cs_certdist OUTER-JOIN
```

If you specify the OUTER-JOIN option, you must also specify the OUTER-JOIN option in all succeeding Record phrases of the query to obtain a left outer join. That is, for multiple Record phrases, all joins in the query following your first left outer join must also be left outer joins. Otherwise, the result is an inner join for all records up to the last inner join in the query.

The OUTER-JOIN option is supported only in Record phrases specified after the first Record phrase in the query. If you specify OUTER-JOIN, you must also specify the OF option, WHERE option, or any combination of the OF and WHERE options. These options are required to select record (the right-most table) for the specified left outer join.

```
FOR EACH customer,
FIRST order OUTER-JOIN OF customer
WHERE order.order-num <50,
FIRST order-line OUTER-JOIN OF order WHERE order-line.item-num < 15
```

This query specifies a left outer join between customer and order, and also between that join and order-line. Thus, for each customer record that has no orders or has no orders with an order-num less than 50, the query returns the customer fields and ? for all fields of the order and order-line tables. In addition, if there are no order-line records with item-num less than 15 for any selected customer and order, the query returns ? for all fields of order-line. Otherwise, it returns each customer record along with its first selected order record and order-line record.

In all statements where multiple Record phrases are allowed, the default join (without the OUTER-JOIN option) is an inner join between record and the table (or join) specified by the previous Record phrase(s). An inner join returns the records selected for the table (or join) on the left side combined with each selected record from the table on the right (record). For an inner join, no records are returned for the table (or join) on the left for which no record is selected from the table on the right (record).

If you specify a Record phrase as an inner join, the current Record phrase and all preceding Record phrases in the query participate in contiguous inner joins, even if prior Record phrases specify the OUTER-JOIN option. Thus, for multiple Record phrases, all joins in the query up to the right-most inner join result in contiguous inner joins.

## 3.6 Buffers

There are circumstances when you may wish to access a single table multiple times in a query. The method for doing this is to use buffers in your query.

Using buffers in a Body Query.

The buffers MUST be named xxn<table> where n is a letter.

Example Syntax:

```
FOR EACH vp_wbsdef WHERE vp_wbsdef.kco = {kco}
AND CAN-DO('COMP,RES,EXCH',vp_wbsdef.vwb_sysstat),
EACH hs_purchaser WHERE hs_purchaser.kco = {kco}
AND hs_purchaser.hev_type = 'SALES'
AND hs_purchaser.hsp_key1 = vp_wbsdef.job_num
AND hs_purchaser.hsp_key2 = vp_wbsdef.jph_phase
AND hs_purchaser.hsp_key = vp_wbsdef.vwb_code
AND hs_purchaser.hca_rewf = 0,
EACH xxahs_purchaser WHERE xxahs_purchaser.kco = hs_purchaser.kco
AND xxahs_purchaser.hev_type = 'VISITOR'
AND xxahs_purchaser.ppp_intref = hs_purchaser.ppp_intref OUTER-JOIN
```

The buffers are automatically created (and deleted) and you can refer to fields using xxn<table>.field

## 3.7 Calculation Programs

This section describes the new capability for the user to store programs (effectively user defined functions) for use in calculations throughout COINS.

The standard COINS syntax is used in the stored program and it can take input parameters from the calculation from which it is called.

User defined functions can call other user defined functions recursively.

The variables used and set in the original calculation and all the user defined functions are globally shared. So a variable set in a user defined program is then available in the calculations that follow it.

### 3.7.1 Creating Calculation Programs

You can define user defined programs using the Calculation Programs maintenance function in System > Misc Maintenance (also available via OA Reporting and BI > OA and BI Setups).

To create a new program, click

- The program name is unique and is the code by which the function will be run when used in a calculation.
- The description and notes are for information only, they do not affect the use and are there for you to document what the function does and how it should be used.
- The calculation code field is the calculation that will run when the program is called. This can contain {n} where n is a number. These will be replaced with the values of the parameters passed to the function at run time. The calculation code can run other user defined calculation programs in the same way that any other calculation.

If the program is run without passing the required {} parameters then they will be replaced in the calculation with blanks.

Variables defined before the calculation is run can be accessed. Variables defined or updated in the calculation program will be available following the call to the calculation program.

### 3.7.2 Calling Calculation Programs

The calculation programs can be called by any calculation in COINS e.g. report field calculations or workflow initialisation calculations etc.

The calculations defined can be run by using the new calculation keyword RUN or RUN\$.

RUN runs a calculation program returning the last decimal value from that calculation whereas RUN\$ expects a string to be returned.

If a calculation program called CALCTEST is defined with the following calculation code:

```
theSum={1} + {2}
```

then it can be run in a calculation as follows:

```
run('CALCTEST','5','7')
```

this would return the answer 12 and could take part in a further calculation e.g.

```
grandTotal = 10 + run('CALCTEST','5','7') + 8
```

this would return 30

The calls to the programs can also use variables

```
A=5
```

```
B=7
```

```
$PROG="CALCTEST"
```

```
Run(PROG,A,B)
```

```
theSum
```

This would also return 12.

By constructing appropriate functions and/or programs in this way it will make using complex features much easier to maintain by placing the complexity in a reusable function.



## 4 OA Calculation Syntax

This section explains the syntax of the various types of calculations that may be used within the COINS OA Framework

## 4.1 Finding the Period Number from a Date

### For Character results:

```
$a=method$(glp-rsp.FdatetoPeriod,Offset,Date)
```

```
$a=method$(glp-rsp.FdatetoMonth,Offset,Date)
```

```
$a=method$(glp-rsp.FdatetoYear,Offset,Date)
```

### For numeric results:

```
b=method(glp-rsp.FdatetoPeriod,Offset,Date)
```

```
b=method(glp-rsp.FdatetoMonth,Offset,Date)
```

```
b=method(glp-rsp.FdatetoYear,Offset,Date)
```

## 4.2 syuquery aggregate

`syuquery.aggregate` and `syuquery.aggregateRSP` (via `rsp`) takes two parameters, query and field list.

Fields in the field list are aggregated. First total is returned (typically might be only one field)

You can then get the other totals with `syuquery.getDAgg,n` and count with `getIAgg`

Multiple fields with comma delimited allowed on last part of method then you can call with `method('syuquery.getDAgg',1); method('syuquery.getDAgg',2);` etc

**Figure 1: Example using `syuquery.aggregate`**

```
method('syuquery.aggregateRSP',"for EACH pp_planenroll WHERE pp_planenroll.ppo_seq =  
" + pp_organisation.ppo_seq + " AND hbp_plan = 'Office Vacation','RO_Accrual_Balance');  
dvacav=method('syuquery.getDAgg',1);  
$string$(dvacav,"->>>, >>>, >>9");
```

**Figure 2: Example using `syuquery.aggregateRSP`**

```
method('syuquery.aggregate',"for EACH se_orderlive WHERE se_orderlive.kco = " + co_  
config.kco + ",first se_order WHERE se_order.sso_intref = se_orderlive.sso_intref and se_  
order.sso_complete = yes,first ar_custaddr OUTER-JOIN WHERE ar_custaddr.kco = se_  
order.kco AND ar_custaddr.rcm_num = se_order.rcm_num AND ar_custaddr.rca_code = se_  
order.rca_code",'kco');  
method('syuquery.getDAgg',1);  
method('syuquery.getIAgg');
```



## 4.3 Performing Calculations on Numerical Values

### 4.3.1 A Simple Count

#### Creating a Simple Count and Accumulating the Calculation Manually

To simply count the number of records being returned, assign the value in each to be 1 – this value can then be totalled and manipulated in the same way as other values. For example – the calculation below would count each line and complete a running total :

```
a = 1; counta = counta + a;
```

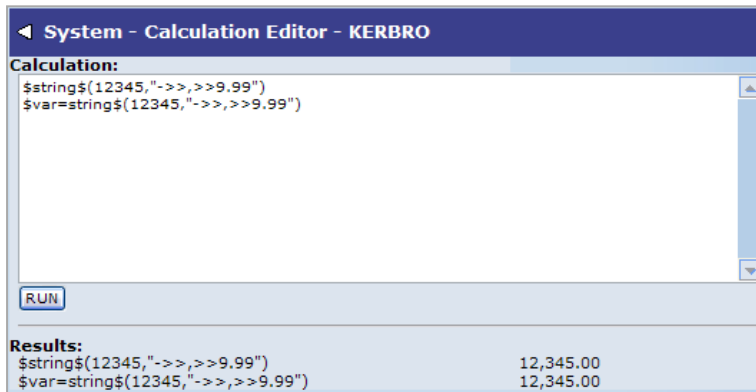
Counts can also be used in conjunction with all other functions to allow counts of certain types of information, e.g. the example below where dStockItemCount is checking the PO Line type and assign 1 if it is a Stock Item, this value is then being totalled.

```
dStockItemCount = INLIST("{po_line.pol_type}","I");  
dTotStockItemCount = dTotStockItemCount + dStockItemCount;
```

### 4.3.2 String\$

- Purpose:** To display a numerical value as a character
- Syntax:** `$string$(number value,"Format")`
- Number** The numeric value to be converted value
  - Format:** The output format required (uses standard format notation and must be surrounded by double quotes)

Figure 3: Example



### 4.3.3 IF

An IF statement is split into two separate areas and all parameters of the 'If statement' are separated with commas

The Test

The Result

#### The TEST

The first section of an 'if statement' can be referred to as the 'Test' it has to compare two numerical values. If the field that is being tested is a character value then that field has to be converted into a numerical value to do the 'Test'. (see calculations on character values)

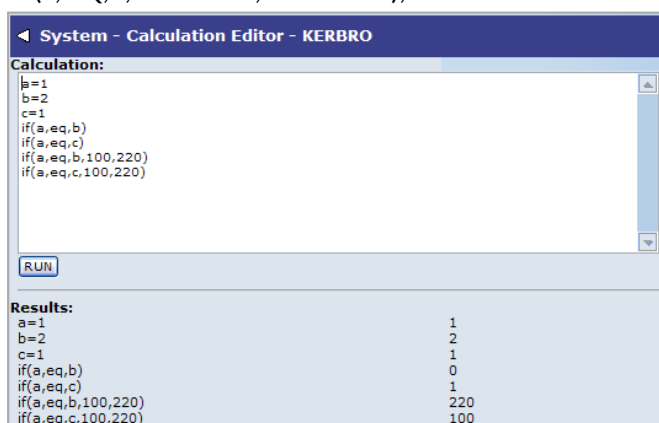
#### The RESULT

The result will either be a true or false answer. Without the specification of any true and false results an if statement will automatically return a 1 if true and a 0 if false. Therefore it may be a simple test between two numbers and can be defined as follows:

**Figure 4: Simple examples of an 'IF statement'.**

IF(a,EQ,b); returns 1 if true / 0 if false

IF(a,EQ,b,truevalue,falsevalue); returns truevalue if true /falsevalue if false.



The operands available for use within the IF function are :

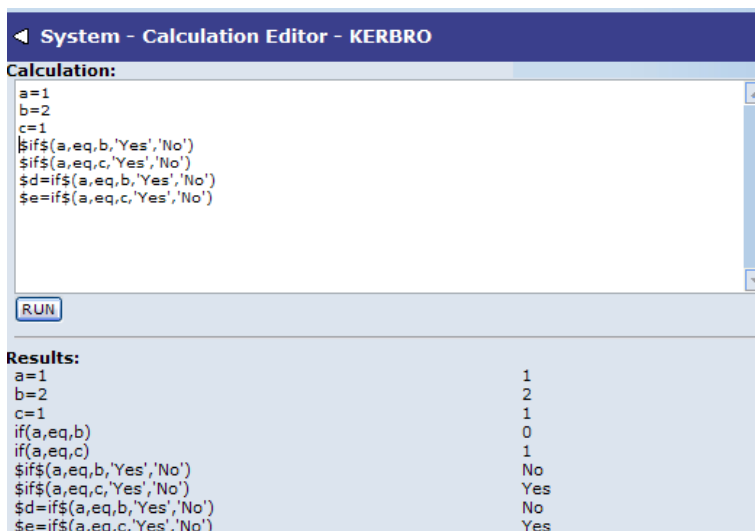
EQ	Equal To
NE	Not Equal To
LT	Less Than
GT	Greater Than
LE	Less Than or Equal To
GE	Greater Than or Equal To

If the 'Result' of an IF statement needs to be a character value then the IF statement must be encapsulated within \$ signs.

`$IF$(a,eq,b,"Yes","No")`

If a variable is to be assigned then the variable will be situated between the first \$ sign and the IF statement.

`$var=IF$(a,eq,b,"Yes","No")`



NB : Using an IF statement to create a simple Flag, in the example below the calculation is setting a variable, dFlag, to either 1 or 0 depending on the comparison of the Valuation Due Date against the Report Selection Date passed through to the report - this flag can be used to multiply values in other fields.



dFlag = IF(datestring(cs\_valcert.cvc\_duedate),LT,datestring('{RS\_date\_\_2}'),1,0);

Note : Date format fields require special consideration – please see below.

Example of above:

Field1={RO\_ContractCosts^TD} \* dFlag

Field2={RO\_ContractRevenue^TD} \* dFlag

Field3=({RO\_ContractRevenue^TD} - {RO\_ContractCosts^TD}) \* dFlag

This means that only when the dFlag is true will these figures for Field1,Field2 & Field 3 will appear as a multiplication by Zero will return value of Zero.

#### 4.3.4 RANGE

These two functions will allow the comparison of data against series of other values:

The RANGE function will return 1 or 0 for a true or false result when comparing a value against an upper and lower limit and should be used in the format:

RANGE(value,lowerlimit,upperlimit)

Therefore:

RANGE(5,1,10)would return 5 (true returns value)

RANGE(5,10,20)would return 0 (false)

In the example below, the RANGE function has been used within an IF function to calculate values in Ageing columns for Purchase Ledger invoices, where the variable Days has been calculated as an integer value of the days between TODAY and the Due Date on the Invoice:

AgeDays = (TODAY - datestring(ap\_invoice.ain\_duedt))

IF(AgeDays,LE,30,ap\_invoice.ain\_balance,0);

IF(RANGE(AgeDays,31,60),GT,1,ap\_invoice.ain\_balance,0);

IF(RANGE(AgeDays,61,90),GT,1,ap\_invoice.ain\_balance,0);

IF(AgeDays,GT,90,ap\_invoice.ain\_balance,0);



◀ System - Calculation Editor - KERBRO

**Calculation:**

```
$ap_invoice.ain_duedt='22/09/10'  
ap_invoice.ain_balance=1000  
AgeDays = (TODAY - datestring(ap_invoice.ain_duedt))  
IF(AgeDays,LE,30,ap_invoice.ain_balance,0);  
IF(RANGE(AgeDays,31,60),GT,1,ap_invoice.ain_balance,0);  
IF(RANGE(AgeDays,61,90),GT,1,ap_invoice.ain_balance,0);  
IF(AgeDays,GT,90,ap_invoice.ain_balance,0);
```

**RUN**

**Results:**

\$ap_invoice.ain_duedt='22/09/10'	22/09/10
ap_invoice.ain_balance=1000	1000
AgeDays = (TODAY - datestring(ap_invoice.ain_duedt))	47
IF(AgeDays,LE,30,ap_invoice.ain_balance,0)	0
IF(RANGE(AgeDays,31,60),GT,1,ap_invoice.ain_balance,0)	1000
IF(RANGE(AgeDays,61,90),GT,1,ap_invoice.ain_balance,0)	0
IF(AgeDays,GT,90,ap_invoice.ain_balance,0)	0

### 4.3.5 Limit

The LIMIT function will return values dependant on where the value falls in relation to the lower and upper limit values. The limit function is useful is reports scores where you cannot allow scores to fall outside the limit of valid scores eg 1 to 10. The format for using LIMIT is:

LIMIT(value,lowerlimit,uperlimit)

If the value is below the lower limit, the function returns the lower limit value, if it is within the range it returns the value itself, if above the upper limit it will return the upper limit value.

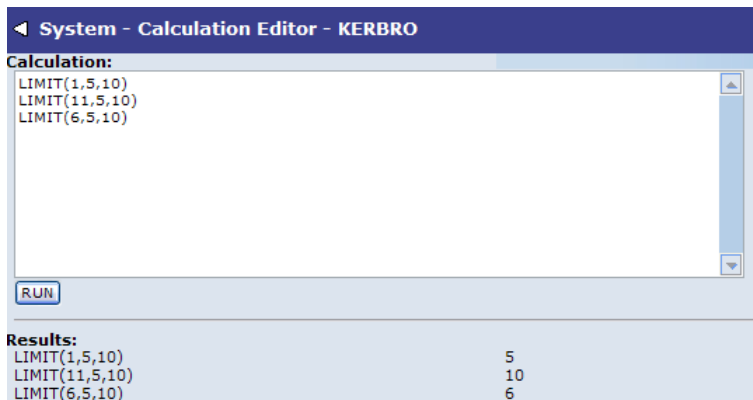
Example

LIMIT(X,A,B) would return x if x is in the range a-b, otherwise return a if x<a, or b if x>b.

LIMIT(1,5,10) would return 5 (FALSE returns lowerlimit)

LIMIT(11,5,10)would return 10 (FALSE returns upperlimit)

LIMIT(6,5,10) would return 6 (TRUE returns value)





### 4.3.6 Max

`max(a,b,c...)` returns the value of the largest entry.



### 4.3.7 Min

min(a,b,c...) returns the value of the smallest entry.

### 4.3.8 Sum

sum(a,b,c...) adds all the entries.

### 4.3.9 non-zero

nonzero(a,b,c) returns 0 if all entries equal zero. Returns 1 if any entry is non-zero.

Nonzero() is a very useful calculation to use in the Calculate Condition which will execute as soon as a record has been read. Only when the value is Non zero will the record display on the report. This will eliminate records where all values are zero.

Example of use: If a report is showing three records for each Contract - Revenue, Costs & Profit.

QUERY:- FOR EACH jc\_job WHERE jc\_job.kco = {kco}

If we want to suppress records where for a contract all three of these fields are zero. Include a Calculate condition with the following syntax:

```
nonzero(jc_job.ContractCosts^TD,jc_job.RO_ContractRevenue^TD,jc_job.RO_ContractRevenue^TD - jc_job.RO_ContractCosts^TD);
```

NB. You must qualify out fields with table names in the Calculation Field



### 4.3.10entry

entry(n,a,b,c) returns the value of the n<sup>th</sup> entry in the list a,b,c. Returns 0 if n is outside the number of entries in the list.





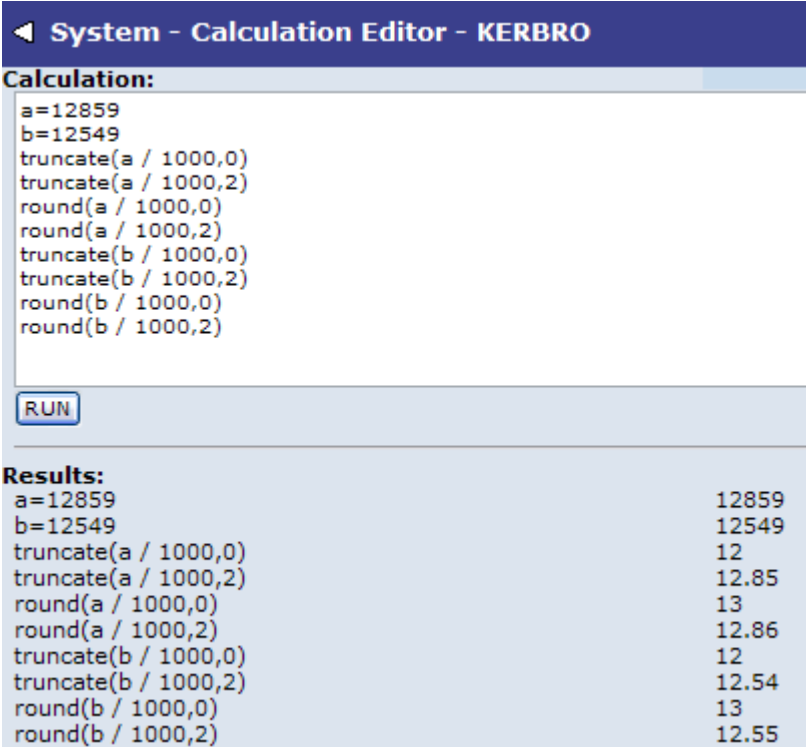
### 4.3.1 truncate

truncate(a,n) truncates a to n decimal places

.

### 4.3.12round

round(a,n) : rounds a to n decimal places.



**System - Calculation Editor - KERBRO**

**Calculation:**

```
a=12859
b=12549
truncate(a / 1000,0)
truncate(a / 1000,2)
round(a / 1000,0)
round(a / 1000,2)
truncate(b / 1000,0)
truncate(b / 1000,2)
round(b / 1000,0)
round(b / 1000,2)
```

**RUN**

**Results:**

a=12859	12859
b=12549	12549
truncate(a / 1000,0)	12
truncate(a / 1000,2)	12.85
round(a / 1000,0)	13
round(a / 1000,2)	12.86
truncate(b / 1000,0)	12
truncate(b / 1000,2)	12.54
round(b / 1000,0)	13
round(b / 1000,2)	12.55



## 4.4 String Calculations

This section details the syntax for string (character) value calculations.

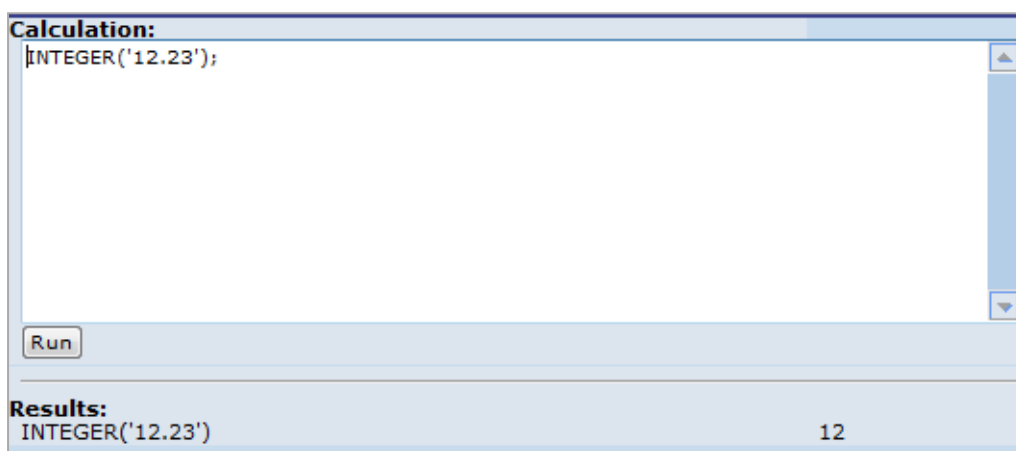
### 4.4.1 INTEGER

Purpose: Converts a character string of numbers into an integer

Syntax INTEGER('string')

string A quoted string of numeric characters

Figure 5: Example of Integer Calculation



## 4.4.2 DECIMAL

Purpose: Converts a character string of numbers into an decimal

Syntax DECIMAL('string')

string A quoted string of numeric characters

Figure 6: Example



### 4.4.3 INLIST

An IF statements can perform comparisons on numerical values only, therefore when the Testing field is a character it is necessary to turn the value into a number so that a comparison can be made. The INLIST function can be used to perform as such on character values. INLIST must be used is making an exact match and should be used in the format :

```
INLIST("field1","Test1,Test2")
```

It will return 1 for a true result and 0 for a false result.

Therefore

```
INLIST("a","a,b,c")would return 1
```

```
INLIST("a","b,c")would return 0
```

#### Using INLIST to assign text values to a column

The following example checks the database field on a Purchase Order Header which is a logical field and instead of returning a TRUE or FALSE value will return either E-Commerce or Standard

```
$IF$(INLIST(po_hdr.poh_xmtd,"yes"),EQ,1,"E-Commerce","Standard");
```

In the next example (Where po\_hdr.poh\_mpo is the type of the Purchase Order -for example, M - Material, P - Plant, O - Overhead, I - Stock, W - Workshop, A - Asset, S - Subcontract), the type of PO is checked against a list of possibilities and the test Material or Trade is output

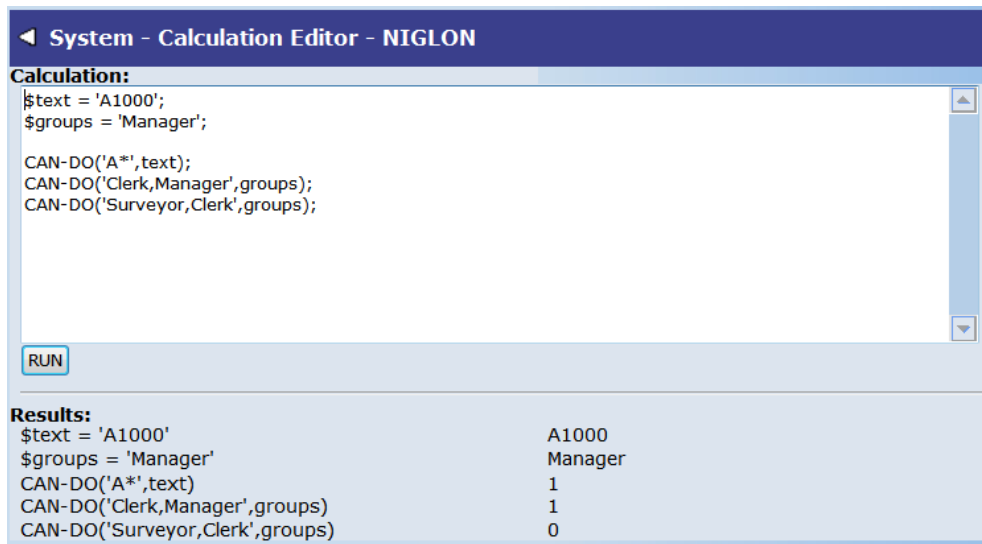
```
$IF$(INLIST(po_hdr.poh_mpo,"M,P,O,I,W,A"),EQ,1,"Material","Trade");
```

The inlist statement becomes equivalent to the 'a' variable in the Test sections of the 'if statement' (See Reference earlier in documentation)as this will result in value of either 1 or zero it can now be tested within the numeric test. The \$ signs are around the \$IF\$ purely because the results of the 'if statement' are characters.

#### 4.4.4 CAN-DO

Purpose:	The CAN-DO function tests a value against a comma separated list of criteria in a similar way to INLIST but will allow the use of wildcards and exclusions. It should be used in the format
Syntax:	CAN-DO(test,comparison)  Returns 1 if True, 0 if False
test:	A single value, comma separated list, or wildcards such as '*' for multi-character tests or '!' for exclusions.
comparison:	A string, string variable or character field against which the test will be compared

Figure 7: Example



**System - Calculation Editor - NIGLON**

**Calculation:**

```
$text = 'A1000';
$groups = 'Manager';

CAN-DO('A*',text);
CAN-DO('Clerk,Manager',groups);
CAN-DO('Surveyor,Clerk',groups);
```

**RUN**

**Results:**

\$text = 'A1000'	A1000
\$groups = 'Manager'	Manager
CAN-DO('A*',text)	1
CAN-DO('Clerk,Manager',groups)	1
CAN-DO('Surveyor,Clerk',groups)	0

## 4.4.5 MASK

The MASK function will return a 1 or 0 for a true or false result when comparing a value against a combination of a RANGE of values and a CAN-DO list and should be used in the format :

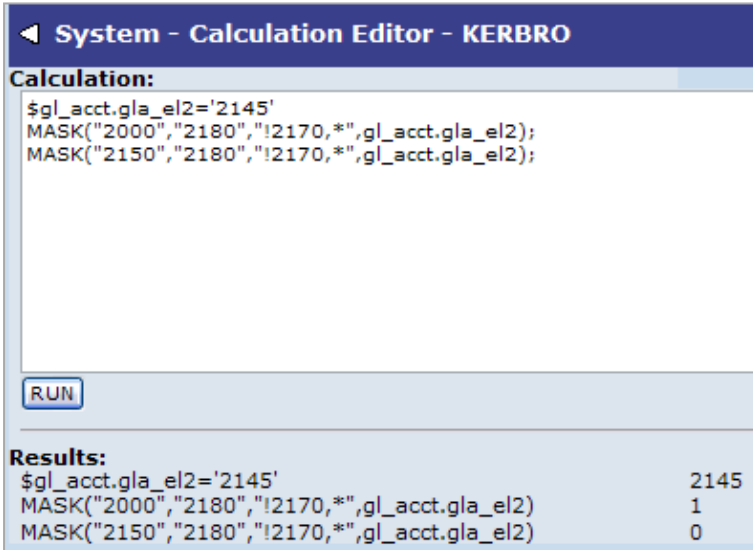
MASK(a,b,c,d)

Where a is the range from, b the range to, c is a can-do list and d the value to be masked. In effect this function behaves as a From, To, List selection in COINS.

For example to check a GL Account - element 2 within a range from 2000 to 2180 but excluding 2170 :

```
MASK("From","To","Matches","Field");
```

```
MASK("2000","2180","!2170,*",gl_acct.gla_el2);
```



**System - Calculation Editor - KERBRO**

**Calculation:**

```
$gl_acct.gla_el2='2145'  
MASK("2000","2180","!2170,*",gl_acct.gla_el2);  
MASK("2150","2180","!2170,*",gl_acct.gla_el2);
```

**RUN**

**Results:**

\$gl_acct.gla_el2='2145'	2145
MASK("2000","2180","!2170,*",gl_acct.gla_el2)	1
MASK("2150","2180","!2170,*",gl_acct.gla_el2)	0



## 4.4.6 ENTRY\$


**Purpose:** Returns a character string entry from a list based on an integer position.

**Syntax:** ENTRY\$(element,'list','delimiter')

**element:** An integer value that corresponds to the position of a character string in a list of values. If the value of the element does not correspond to an entry in the list, an error condition will occur.

**list:** A list of character strings. Separate entries with a delimiter – usually a comma and surround the list with single quotes.

**delimiter:** A delimiter you define for the list. This allows ENTRY\$ to operate on non-comma separated lists. The delimiter must be a single character. Surround the delimiter with single quotes.



The screenshot shows a software window titled "System - Calculation Editor - NIGLON". It contains a "Calculation:" section with a text area containing the following code:

```
$ENTRY$(3,'A,B,C,D','');  
$ENTRY$(2,'E-F-G','-');  
$email = 'nigel.longley@coins-global.com';  
$ENTRY$(1,email,'@');  
$ENTRY$(2,email,'@');
```

Below the code is a "RUN" button. The "Results:" section displays the output of the calculations:

\$ENTRY\$(3,'A,B,C,D','')	C
\$ENTRY\$(2,'E-F-G','-')	F
\$email = 'nigel.longley@coins-global.com'	nigel.longley@coins-global.com
\$ENTRY\$(1,email,'@')	nigel.longley
\$ENTRY\$(2,email,'@')	coins-global.com

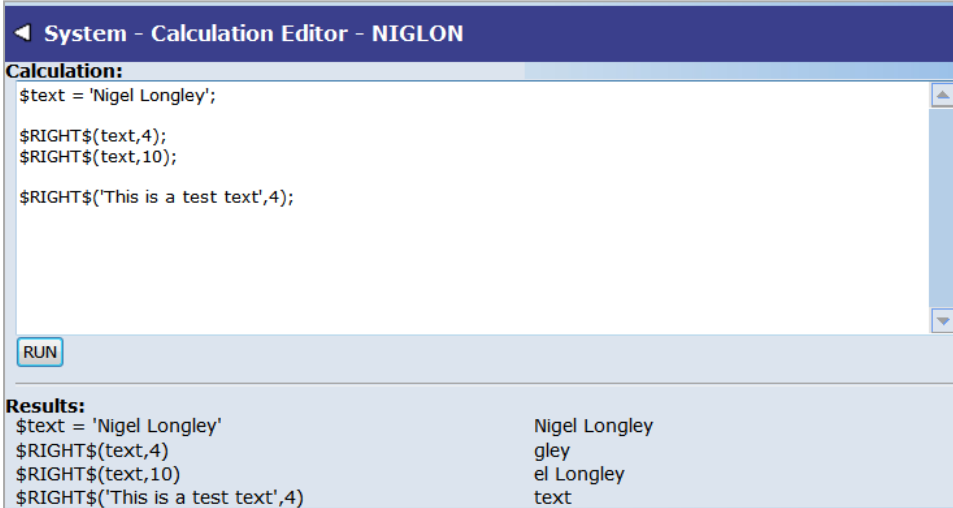
## 4.4.7 RIGHT\$

**Purpose:** Returns a specified number of characters from a character string starting from the last character position.

**Syntax:** RIGHT\$(string,integer)

**string:** A character expression. The string may be a constant, field name, variable, or expression that results in a character value.

**integer:** The number of characters to be returned



The screenshot shows a window titled "System - Calculation Editor - NIGLON". The "Calculation:" section contains the following code:  
\$text = 'Nigel Longley';  
\$RIGHT\$(text,4);  
\$RIGHT\$(text,10);  
\$RIGHT\$('This is a test text',4);

Below the code is a "RUN" button. The "Results:" section displays the output of the calculations:

\$text = 'Nigel Longley'	Nigel Longley
\$RIGHT\$(text,4)	gley
\$RIGHT\$(text,10)	el Longley
\$RIGHT\$('This is a test text',4)	text

Example:

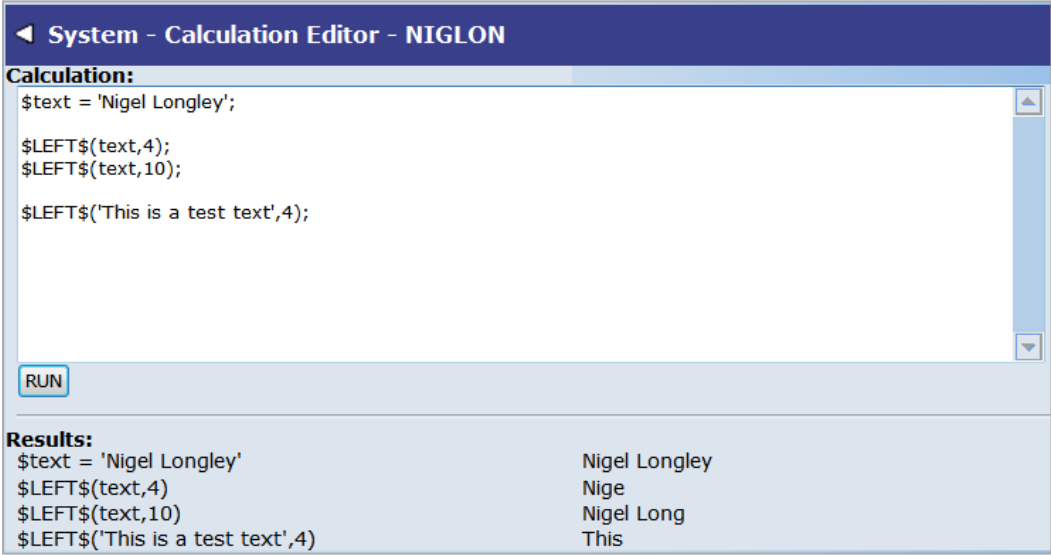
## 4.4.8 LEFT\$

**Purpose:** Returns a specified number of characters from a character string starting from the first character position.

**Syntax:** LEFT\$(string,integer)

**string:** A character expression. The string may be a constant, field name, variable, or expression that results in a character value.

**integer:** The number of characters to be returned



**System - Calculation Editor - NIGLON**

**Calculation:**

```
$text = 'Nigel Longley';  
  
$LEFT$(text,4);  
$LEFT$(text,10);  
  
$LEFT$('This is a test text',4);
```

**RUN**

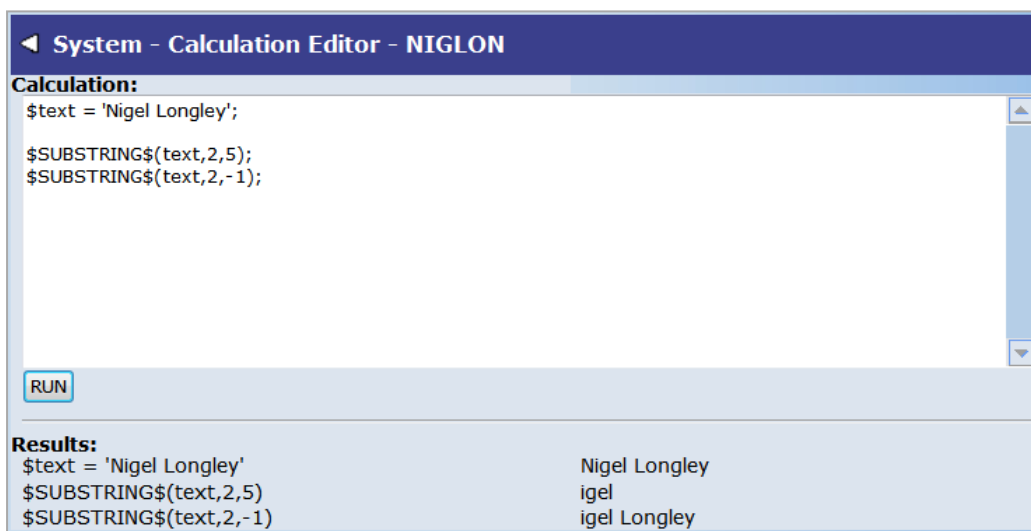
**Results:**

\$text = 'Nigel Longley'	Nigel Longley
\$LEFT\$(text,4)	Nige
\$LEFT\$(text,10)	Nigel Long
\$LEFT\$('This is a test text',4)	This

**Example:**

## 4.4.9 SUBSTRING\$

- Purpose:** Extracts a portion of a character string from a field or variable
- Syntax:** SUBSTRING\$(source,position,length)
- source:** A character expression. The string may be a constant, field name, variable, or expression that results in a character value.
- position:** An integer expression that indicates the position of the first character you want to extract from the source
- length:** An integer that indicates the number of character you want to extract from the source. If you specify -1 as the length, SUBSTRING\$ uses the remainder of the string from the specified position



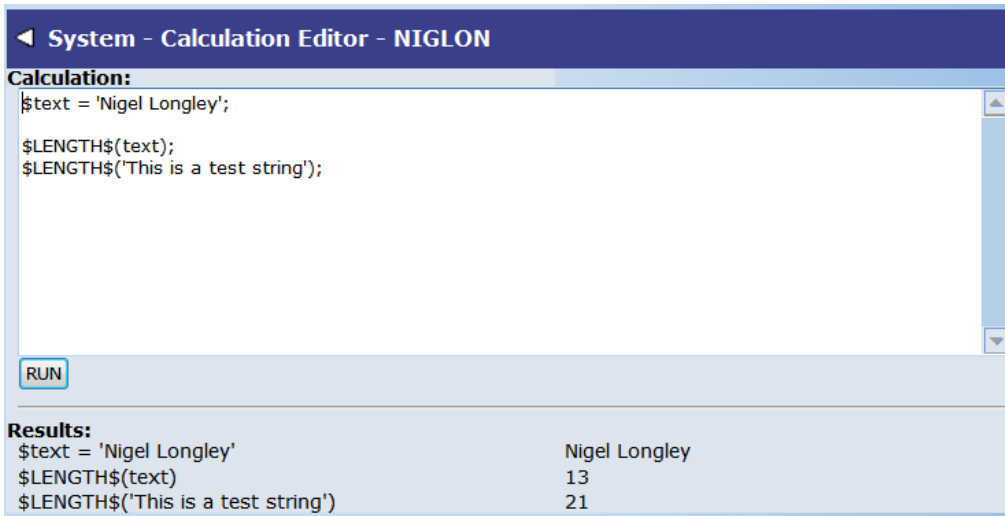
Example:

### 4.4.10LENGTH\$

Purpose: Returns the number of characters in a string

Syntax: LENGTH\$(string)

string: A character expression. The string may be a constant, field name, variable, or expression that results in a character value.



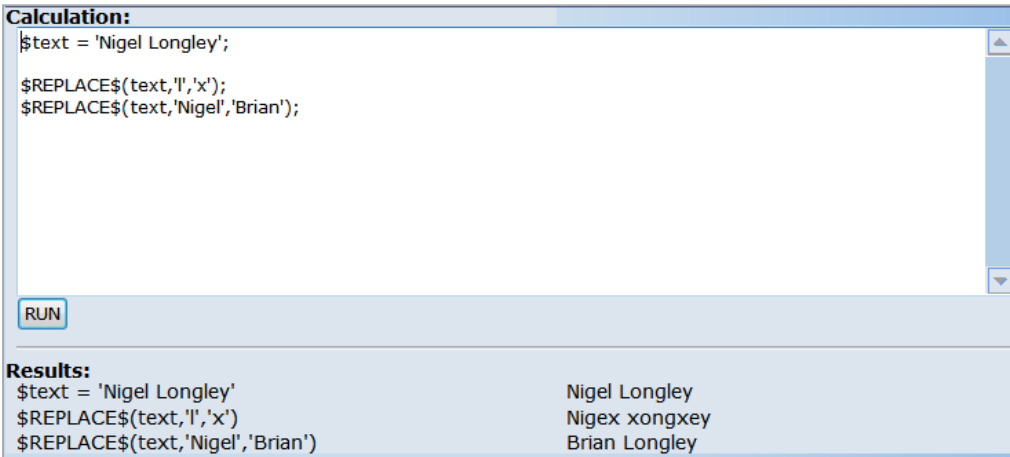
The screenshot shows a software window titled "System - Calculation Editor - NIGLON". It contains a "Calculation:" section with a text area containing the following code:  
\$text = 'Nigel Longley';  
\$LENGTH\$(text);  
\$LENGTH\$('This is a test string');  
Below the text area is a "RUN" button. At the bottom, a "Results:" section displays the output of the calculations in a table format.

Calculation	Result
\$text = 'Nigel Longley'	Nigel Longley
\$LENGTH\$(text)	13
\$LENGTH\$('This is a test string')	21

Example:

### 4.4.1 REPLACE\$

- Purpose:** Replaces characters in a string
- Syntax:** REPLACE\$(string,original,replacement)
- string:** A character expression. The string may be a constant, field name, variable, or expression that results in a character value.
- original** The characters in the string to be replaced
- replacement** The replacement characters



**Calculation:**

```
$text = 'Nigel Longley';  
$REPLACE$(text,'l','x');  
$REPLACE$(text,'Nigel','Brian');
```

**RUN**

**Results:**

\$text = 'Nigel Longley'	Nigel Longley
\$REPLACE\$(text,'l','x')	Nigex xongxey
\$REPLACE\$(text,'Nigel','Brian')	Brian Longley

Example:

### 4.4.12 CAPS\$

Purpose:	Converts any characters in a character string to uppercase or lowercase characters, and returns the resulting string.
Syntax:	CAPS\$(string,flag)
	string: A character expression. The string may be a constant, field name, variable, or expression that results in a character value.
	flag: 1 – String will be converted to uppercase 2 – String will be converted to lowercase

Figure 8: Example

**Calculation:**

```
$text = 'Nigel Longley';
$a = CAPS$(text,1);
$b = CAPS$(text,0);
```

▲

▼

---

**Results:**

<pre>\$text = 'Nigel Longley' \$a = CAPS\$(text,1) \$b = CAPS\$(text,0)</pre>	<pre>Nigel Longley NIGEL LONGLEY nigel longley</pre>
---	--

### 4.4.13 TRIM\$

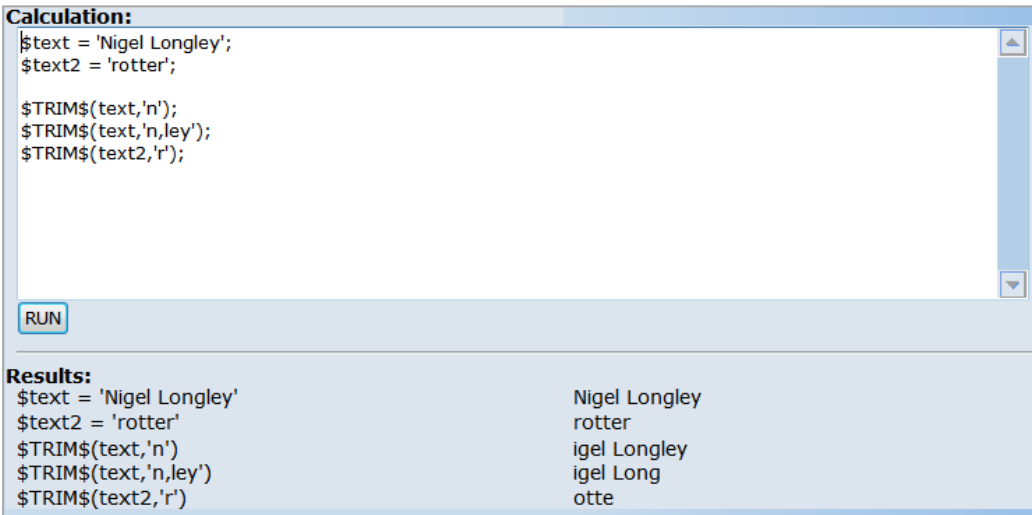
**Purpose:** Removes leading and trailing specified characters from a character string

**Syntax:** TRIM\$(string,trim-chars)

**string:** A character expression. The string may be a constant, field name, variable, or expression that results in a character value.

**trim-** A character expression that specifies the characters to trim from the string.

**chars:** A single character expression will be removed from both beginning and end of string, a comma separated trim-chars will specify different beginning and end characters.



**Calculation:**

```
$text = 'Nigel Longley';  
$text2 = 'rotter';  
  
$TRIM$(text,'n');  
$TRIM$(text,'n,ley');  
$TRIM$(text2,'r');
```

**Results:**

\$text = 'Nigel Longley'	Nigel Longley
\$text2 = 'rotter'	rotter
\$TRIM\$(text,'n')	igel Longley
\$TRIM\$(text,'n,ley')	igel Long
\$TRIM\$(text2,'r')	otte

**Example:**

Note: There is a Right-Trim\$ command to remove trailing characters from a character string, but no Left-Trim\$ command. Therefore for a calculation to convert say '020' to '20' the following solution is suggested

TRIM\$('020' + ' ','0') which would give you '20 '

Or

TRIM\$(TRIM\$('020' + ' ','0'),' ') to get rid of the trailing space if required






### 4.4.14 RIGHT-TRIM\$

**Purpose:** Removes trailing specified characters from a character string

**Syntax:** RIGHT-TRIM\$(string,trim-chars)

**string:** A character expression. The string may be a constant, field name, variable, or expression that results in a character value.

**trim-chars:** A character expression that specifies the characters to trim from the string.



**Calculation:**

```
$text = 'Nigel Longley';  
$text2 = 'rotter';  
  
$RIGHT-TRIM$(text,'y');  
$RIGHT-TRIM$(text,'ley');  
$RIGHT-TRIM$(text2,'r');
```

**RUN**

**Results:**

\$text = 'Nigel Longley'	Nigel Longley
\$text2 = 'rotter'	rotter
\$RIGHT-TRIM\$(text,'y')	Nigel Longle
\$RIGHT-TRIM\$(text,'ley')	Nigel Long
\$RIGHT-TRIM\$(text2,'r')	rotte

**Example:**

## 4.4.15 INDEX

**Purpose:** Returns an integer that indicates the position of the target string within a source string

**Syntax:** INDEX(source,target)

**source:** A character expression. The string may be a constant, field name, variable, or expression that results in a character value.

**target:** A character expression that specifies the characters to trim from the string.



**Calculation:**

```
$text = 'Nigel Longley';  
$text2 = 'rotter';  
  
INDEX(text,'gel');  
INDEX(text2,'e');
```

**RUN**

**Results:**

\$text = 'Nigel Longley'	Nigel Longley
\$text2 = 'rotter'	rotter
INDEX(text,'gel')	3
INDEX(text2,'e')	5

### 4.4.1 R-INDEX

**Purpose:** Returns an integer that indicates the position of the target string within a source string. In contrast to the INDEX function, R-INDEX performs the search from right to left

**Syntax:** INDEX(source,target)

**source:** A character expression. The string may be a constant, field name, variable, or expression that results in a character value.

**target:** A character expression that specifies the characters to trim from the string.



**Calculation:**

```
$text = 'Nigel Longley';  
$text2 = 'rotter';  
  
R-INDEX(text,'N');  
R-INDEX(text2,'e');
```

**Results:**

\$text = 'Nigel Longley'	Nigel Longley
\$text2 = 'rotter'	rotter
R-INDEX(text,'N')	9
R-INDEX(text2,'e')	5

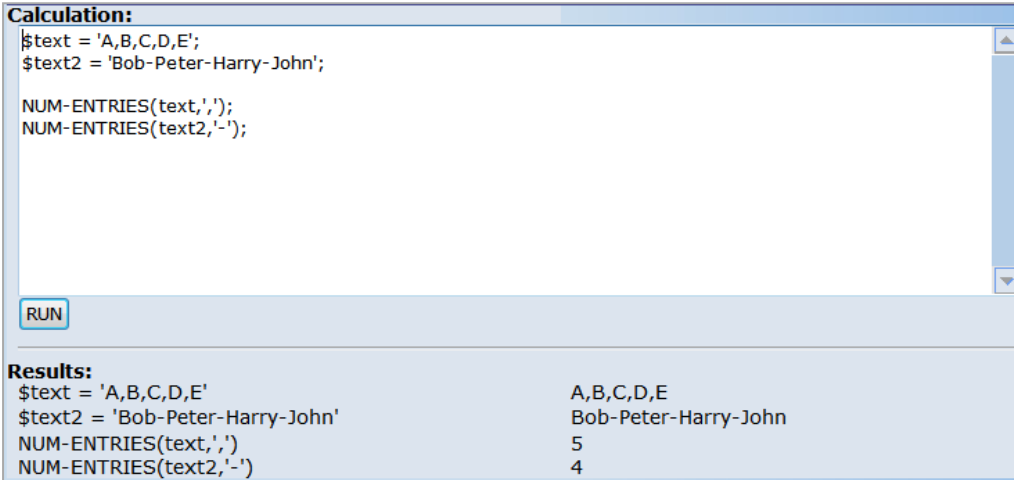
### 4.4.17 NUM-ENTRIES

Purpose: Returns the number of items in a list

Syntax: NUM-ENTRIES(list,delimiter)

list: A list of strings

delimiter: A delimiter you define for the list. The default is a comma. This allows the function to operate on a non-comma separated list.



**Calculation:**

```
$text = 'A,B,C,D,E';  
$text2 = 'Bob-Peter-Harry-John';  
  
NUM-ENTRIES(text,',');  
NUM-ENTRIES(text2,'-');
```

**Results:**

\$text = 'A,B,C,D,E'	A,B,C,D,E
\$text2 = 'Bob-Peter-Harry-John'	Bob-Peter-Harry-John
NUM-ENTRIES(text,',')	5
NUM-ENTRIES(text2,'-')	4

## 4.4.18 LOOKUP

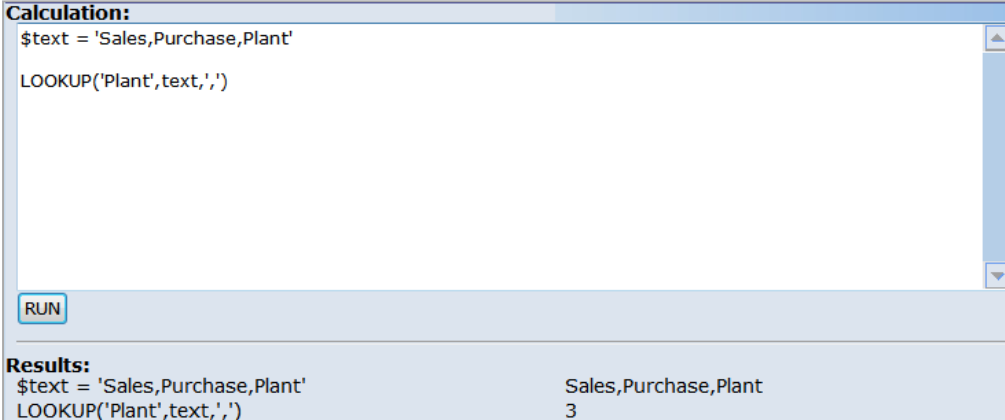
**Purpose:** Returns an integer giving the position of an expression in a list.

**Syntax:** LOOKUP(expression,list,delimiter)

**expression:** A constant, field name, variable name, or expression that results in a character value that you want to look up with a list of character expressions.

**list:** A list of character expressions. Separate each entry in list with a delimiter – the default is a comma.

**delimiter:** A delimiter you define for the list. The default is a comma. This allows the function to operate on a non-comma separated list.



The screenshot shows a software interface for a calculation. The top section is titled "Calculation:" and contains the following text: `$text = 'Sales,Purchase,Plant'` and `LOOKUP('Plant',text,',')`. Below this text is a "RUN" button. The bottom section is titled "Results:" and displays the output of the calculation in two columns. The first column shows the input text: `$text = 'Sales,Purchase,Plant'` and `LOOKUP('Plant',text,',')`. The second column shows the result: `Sales,Purchase,Plant` and `3`.

Results:	
<code>\$text = 'Sales,Purchase,Plant'</code>	<code>Sales,Purchase,Plant</code>
<code>LOOKUP('Plant',text,',')</code>	<code>3</code>

## 4.5 Time & Date Calculations

It is recommended that in all instances where calculations are required on a date, that the date is converted to an integer – then for display purposes converted back to a standard date format.

Once a date is in integer form it can then be used in further calculations (i.e. add 7 to calculate a week later or within an IF statement).



### 4.5.1 date

date(DD,MM,YYYY)Returns the integer value of the date. This is useful if combined with get () or set() for report stores, or for comparison with other dates.





## 4.5.2 datestring

`datestring("dd/mm/yyyy")` Returns the integer value of the date string.



### 4.5.3 weekday

weekday(a)Returns the number of the day in the week (where Sunday=1), if 'a' is the integer value of a date. For example, 11-Feb-04 is a Wednesday, so weekday(date(11,02,2004)) returns 3.



#### 4.5.4 day

day(a)Returns the date in the month if 'a' is the integer value of a date.



## 4.5.5 month

month(a) Returns the number of the month if 'a' is the integer value of a date.



## 4.5.6 year

year(a) Returns the year (as a four-digit integer) if 'a' is the integer value of a date.



## 4.5.7 time

time\$("12345")Would return 03:45:23



## 4.5.8 Weekdays

Weekday(decimal-date1,decimal-date2)returns an integer for the number of weekdays (i.e. excluding weekends) between two dates

## 5 OA and BI Utilities

To assist developers in creating and testing OA Queries and calculations, a number of utilities are available with the OA & BI Reporting Module.

The commonly used utilities are:

- Database Enquiry
- Query Editor
- Calculation Editor
- Object Enquiry



## 5.1 Database Enquiry

To assist users in Open Architecture to understand and exploit the coins database schema when creating enquiries and reports a powerful tool has been developed which provides information on all coins tables, fields and formats.

The Database Enquiry provides detailed information on the structure of every table in the coins database.

Table	Description	DB	ID
<a href="#">abi_dta</a>	ABI Data dta file	coins	abi
<a href="#">abi_dtc</a>	ABI Data dtc file	coins	abc
<a href="#">abi_dtd</a>	ABI Data dtd file	coins	abd
<a href="#">abi_fields</a>	ABI field mapping	coins	afd
<a href="#">abi_load</a>		coins	abi
<a href="#">abi_tran</a>	ABI transaction log	coins	atr
<a href="#">abi_tran_det</a>	ABI transfer details	coins	atd
<a href="#">ac_scheme</a>	Scheme	coins	ack
<a href="#">ac_schtype</a>	Scheme Types	coins	ach
<a href="#">ai_asset</a>	Asset Repository	coins	ais
<a href="#">ap_check</a>	P/L Payment	coins	acs
<a href="#">ap_choose</a>	P/L Open Payment	coins	aco
<a href="#">ap_civline</a>	Capital Invoice Line	coins	aci
<a href="#">ap_confir</a>	Configuration File	coins	act
<a href="#">ap_invdist</a>	Invoice Distribution File	coins	aid
<a href="#">ap_invhist</a>	Invoice History	coins	ahf
<a href="#">ap_invline</a>	P/L Invoice Lines	coins	ail
<a href="#">ap_invoice</a>	P/L Invoice	coins	ain
<a href="#">ap_invopen</a>	P/L Open Invoice	coins	aop
<a href="#">ap_invquery</a>	Invoice queries	coins	aiq
<a href="#">ap_instdist</a>	P/L Intrastat Distribution Line	coins	api
<a href="#">ap_pay</a>	P/L Payment Allocation	coins	apy
<a href="#">ap_pcard</a>	PCards	coins	app
<a href="#">ap_pcardline</a>	PCard Transaction Lines	coins	apd
<a href="#">ap_pcardtran</a>	PCard Transactions	coins	apt

Filters at the top of the page allow you to search for specific tables, fields and table types.

Table	Description	DB	ID
-------	-------------	----	----

It also provides field information, descriptions, formats as well as documentation supplied by the coins Development Team to support users in creating their queries. This includes all calculated fields available via the RSP's (Record Service Procedures).



index	Primary	Unique	Field	Documentation
ain_key	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	kco+ avm_num+ ain_inv+	kco+avm_num+ain_inv+
ain_key1		<input checked="" type="checkbox"/>	kco+ cob_num+ cob_line+	kco+cob_num+cob_line+
ain_key10			kco+ ohd_num+	
ain_key2			kco+ ain_inv+	kco+ain_inv+
ain_key3			kco+ avm_num+ ain_supref+	kco+avm_num+ain_supref+
ain_key4			kco+ ain_supref+ avm_num+	
ain_key5			kco+ avm_num+ ain_entry+ ain_anal+ ain_inv+	
ain_key6			kco+ avm_num+ ain_entry+ ain_anal+ ain_supref+	
ain_key7			kco+ ain_entry+ ain_anal+ ain_inv+	
ain_key8			kco+ avm_num+ ain_idate+ ain_inv+	
ain_key9			kco+ ain_idate+ avm_num+ ain_inv+	

Field	Label	Data Type	Format	Documentation
ain_altcur	Supplier Alternative Currency	logical	yes/no	RW <input checked="" type="checkbox"/> Whether alternative currencies i.e. different to the account currency (ap_vendor_cur_code) are allowed to be entered on the account.
ain_amount	Gross Amount	decimal	->>>>>9.99	DB <input checked="" type="checkbox"/> The gross amount of the Invoice. <input checked="" type="checkbox"/> The gross amount of the invoice in base currency. Equivalent to ain_cur_gross[2].
ain_anal	Analysis (Contract or Dept)	character	x(8)	DB <input checked="" type="checkbox"/> The costing analysis for this invoice. <input checked="" type="checkbox"/> The main contract or department to which the invoice is assigned. Depends on ain_entry.
ain_apacct	P/L Control	character	X(19)	DB <input checked="" type="checkbox"/> The creditor's control account posted to when the invoice was committed.
ain_atotaxpay	ATO Reporting	character	x(4)	DB <input checked="" type="checkbox"/> The option which allows to select behaviour for specific invoice in order to providing correct data to ATO (Australian Taxation Office). It can take following options: <ul style="list-style-type: none"> <li>• Include in the ATO Taxable Payments report</li> <li>• Exclude from the ATO Taxable Payments report</li> <li>• Blank: Default to the Supplier setting or if not set default to the Company workbench setting for this company (check PU/USECIS parameter).</li> </ul>

In addition the Database Enquiry will provide the links available to associated tables and also provide the syntax required to build a query to create these links within Page and Report Designer.

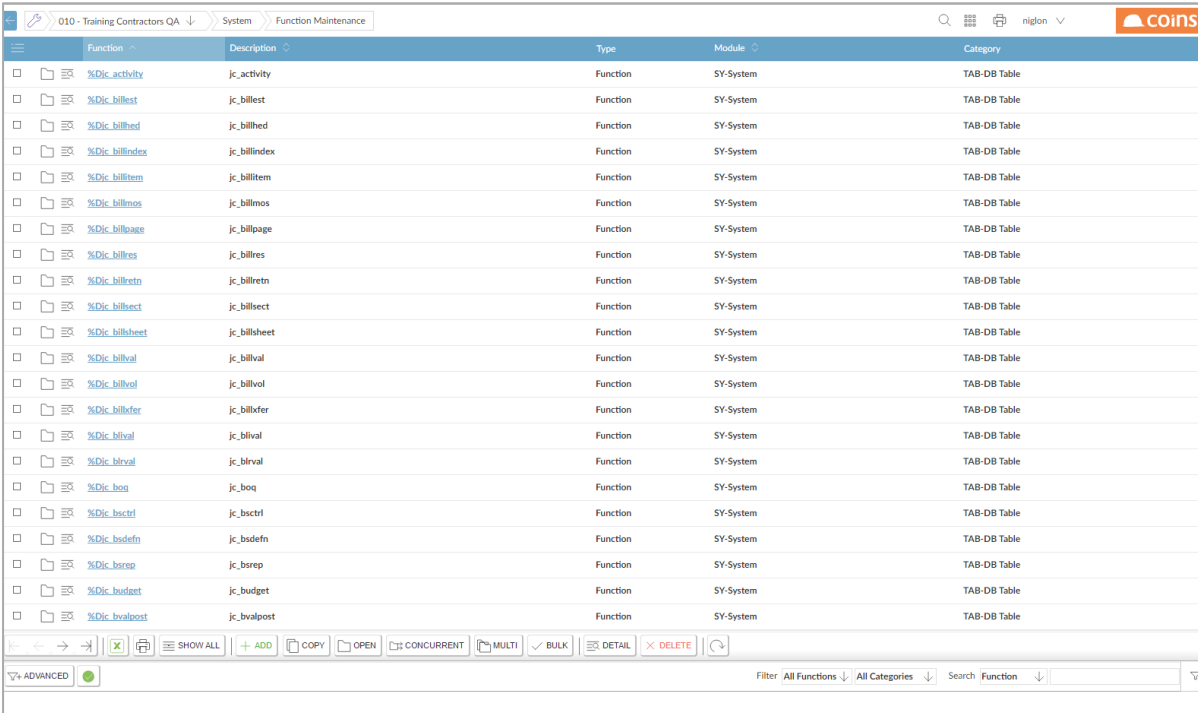
From	To	Join To	Documentation	Code
1	*	ap_cinvline	ap_cinvline OF ap_invoice	ap_cinvline.kco+ap_invoice.kco AND ap_cinvline.avm_num+ap_invoice.avm_num AND ap_cinvline.ain_inv+ap_invoice.ain_inv
1	*	ap_invdist	ap_invdist OF ap_invoice	ap_invdist.kco+ap_invoice.kco AND ap_invdist.avm_num+ap_invoice.avm_num AND ap_invdist.ain_inv+ap_invoice.ain_inv
1	*	ap_invline	ap_invline OF ap_invoice	ap_invline.kco+ap_invoice.kco AND ap_invline.avm_num+ap_invoice.avm_num AND ap_invline.ain_inv+ap_invoice.ain_inv
1	1	ap_invopen	ap_invopen OF ap_invoice	ap_invopen.kco+ap_invoice.kco AND ap_invopen.avm_num+ap_invoice.avm_num AND ap_invopen.ain_inv+ap_invoice.ain_inv
1	1	ap_invquery	ap_invquery OF ap_invoice	ap_invquery.kco+ap_invoice.kco AND ap_invquery.avm_num+ap_invoice.avm_num AND ap_invquery.ain_inv+ap_invoice.ain_inv
1	*	ap_itsdist	ap_itsdist OF ap_invoice	ap_itsdist.kco+ap_invoice.kco AND ap_itsdist.avm_num+ap_invoice.avm_num AND ap_itsdist.ain_inv+ap_invoice.ain_inv
1	*	ap_vatdist	ap_vatdist OF ap_invoice	ap_vatdist.kco+ap_invoice.kco AND ap_vatdist.avm_num+ap_invoice.avm_num AND ap_vatdist.ain_inv+ap_invoice.ain_inv
*	1	ap_vendor	ap_vendor OF ap_invoice	ap_vendor.kco+ap_invoice.kco AND ap_vendor.avm_num+ap_invoice.avm_num
*	1	ap_vendsum	ap_vendsum OF ap_invoice	ap_vendsum.kco+ap_invoice.kco AND ap_vendsum.avm_num+ap_invoice.avm_num
*	1	ar_invoice	ar_invoice OF ap_invoice	ar_invoice.kco+ap_invoice.kco AND ar_invoice.cob_num+ap_invoice.cob_num AND ar_invoice.cob_line+ap_invoice.cob_line
*	1	cb_tdist	cb_tdist OF ap_invoice	cb_tdist.kco+ap_invoice.kco AND cb_tdist.cob_num+ap_invoice.cob_num AND cb_tdist.cob_line+ap_invoice.cob_line
*	1	cb_topen	cb_topen OF ap_invoice	cb_topen.kco+ap_invoice.kco AND cb_topen.cob_num+ap_invoice.cob_num AND cb_topen.cob_line+ap_invoice.cob_line
*	1	co_currency	co_currency OF ap_invoice	co_currency.kco+ap_invoice.kco AND co_currency.cur_code+ap_invoice.cur_code
*	1	co_vat	co_vat OF ap_invoice	co_vat.kco+ap_invoice.kco AND co_vat.vat_code+ap_invoice.vat_code
*	1	cs_certificate	cs_certificate OF ap_invoice	cs_certificate.kco+ap_invoice.kco AND cs_certificate.cob_num+ap_invoice.cob_num AND cs_certificate.cob_line+ap_invoice.cob_line
*	1	fa_invoice	fa_invoice OF ap_invoice	fa_invoice.kco+ap_invoice.kco AND fa_invoice.ain_inv+ap_invoice.ain_inv AND fa_invoice.avm_num+ap_invoice.avm_num
*	1	hs_devsum	hs_devsum OF ap_invoice	hs_devsum.kco+ap_invoice.kco AND hs_devsum.job_num+ap_invoice.job_num AND hs_devsum.jph_phase+ap_invoice.jph_phase
*	1	jk_job	jk_job OF ap_invoice	jk_job.kco+ap_invoice.kco AND jk_job.job_num+ap_invoice.job_num
*	1	jk_jobsum	jk_jobsum OF ap_invoice	jk_jobsum.kco+ap_invoice.kco AND jk_jobsum.job_num+ap_invoice.job_num
*	1	jk_phase	jk_phase OF ap_invoice	jk_phase.kco+ap_invoice.kco AND jk_phase.job_num+ap_invoice.job_num AND jk_phase.jph_phase+ap_invoice.jph_phase
*	1	jk_phsum	jk_phsum OF ap_invoice	jk_phsum.kco+ap_invoice.kco AND jk_phsum.job_num+ap_invoice.job_num AND jk_phsum.jph_phase+ap_invoice.jph_phase

## 5.2 Access to the Database Enquiry and Tables

It is possible to control access to the database enquiry and certain tables in COINS OA.

This is controlled via standard coins OA functions which can be assigned to certain Users. This will control which users will have access to which database tables in the enquiry. <sup>1</sup>

Each table in the COINS database has an associate %D function. For example, the function controlling access to table jc\_job is %Djc\_job



Function	Description	Type	Module	Category
%Djc_activity	jc_activity	Function	SY-System	TAB-DB Table
%Djc_billact	jc_billact	Function	SY-System	TAB-DB Table
%Djc_billhed	jc_billhed	Function	SY-System	TAB-DB Table
%Djc_billindex	jc_billindex	Function	SY-System	TAB-DB Table
%Djc_billitem	jc_billitem	Function	SY-System	TAB-DB Table
%Djc_billmos	jc_billmos	Function	SY-System	TAB-DB Table
%Djc_billpage	jc_billpage	Function	SY-System	TAB-DB Table
%Djc_billres	jc_billres	Function	SY-System	TAB-DB Table
%Djc_billretn	jc_billretn	Function	SY-System	TAB-DB Table
%Djc_billsect	jc_billsect	Function	SY-System	TAB-DB Table
%Djc_billsheet	jc_billsheet	Function	SY-System	TAB-DB Table
%Djc_billval	jc_billval	Function	SY-System	TAB-DB Table
%Djc_billvol	jc_billvol	Function	SY-System	TAB-DB Table
%Djc_billxfer	jc_billxfer	Function	SY-System	TAB-DB Table
%Djc_bilval	jc_bilval	Function	SY-System	TAB-DB Table
%Djc_bilrval	jc_bilrval	Function	SY-System	TAB-DB Table
%Djc_boq	jc_boq	Function	SY-System	TAB-DB Table
%Djc_bsctrl	jc_bsctrl	Function	SY-System	TAB-DB Table
%Djc_bsdefn	jc_bsdefn	Function	SY-System	TAB-DB Table
%Djc_bsrep	jc_bsrep	Function	SY-System	TAB-DB Table
%Djc_budget	jc_budget	Function	SY-System	TAB-DB Table
%Djc_bvalpost	jc_bvalpost	Function	SY-System	TAB-DB Table

To amend Function Access in Coins OA – simply go to Users/Groups (under the Function Security Tab on Function Maintenance) and select the User or Group to be amended and open the the User ID or Group Id and change the access as appropriate.

<sup>1</sup>By default all tables are only initially available to the "ROOT" User Group.

010 - Training Contractors QA >> System >> User Details

User afsmlr      Afsaneh Mirazimi

User afsmlr      Last Login 19/10/15 14:58:53

Name Afsaneh Mirazimi      Debug START STOP

Details Groups Printing COINSpplus Preferences Wiki **User Function Access** User Logging Enquiry Favourites User Views System Alert History

Function	Description	Module	Role Type	Type	Category	Access	Group	Allowed
<input type="checkbox"/>	%Djc_activity	jc_activity	SY	Function	TAB	G-Group	Yes - (root)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	%Djc_billest	jc_billest	SY	Function	TAB	G-Group	Yes - (root)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	%Djc_billhed	jc_billhed	SY	Function	TAB	G-Group	Yes - (root)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	%Djc_billindex	jc_billindex	SY	Function	TAB	G-Group	Yes - (root)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	%Djc_billitem	jc_billitem	SY	Function	TAB	G-Group	Yes - (root)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	%Djc_billmos	jc_billmos	SY	Function	TAB	G-Group	Yes - (root)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	%Djc_billpage	jc_billpage	SY	Function	TAB	G-Group	Yes - (root)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	%Djc_billres	jc_billres	SY	Function	TAB	G-Group	Yes - (root)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	%Djc_billretn	jc_billretn	SY	Function	TAB	G-Group	Yes - (root)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	%Djc_billsect	jc_billsect	SY	Function	TAB	G-Group	Yes - (root)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	%Djc_billsheet	jc_billsheet	SY	Function	TAB	G-Group	Yes - (root)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	%Djc_billval	jc_billval	SY	Function	TAB	G-Group	Yes - (root)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	%Djc_billvol	jc_billvol	SY	Function	TAB	G-Group	Yes - (root)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	%Djc_billxfer	jc_billxfer	SY	Function	TAB	G-Group	Yes - (root)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	%Djc_blval	jc_blval	SY	Function	TAB	G-Group	Yes - (root)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	%Djc_blrval	jc_blrval	SY	Function	TAB	G-Group	Yes - (root)	<input checked="" type="checkbox"/>

SHOW ALL      Set Access to Yes

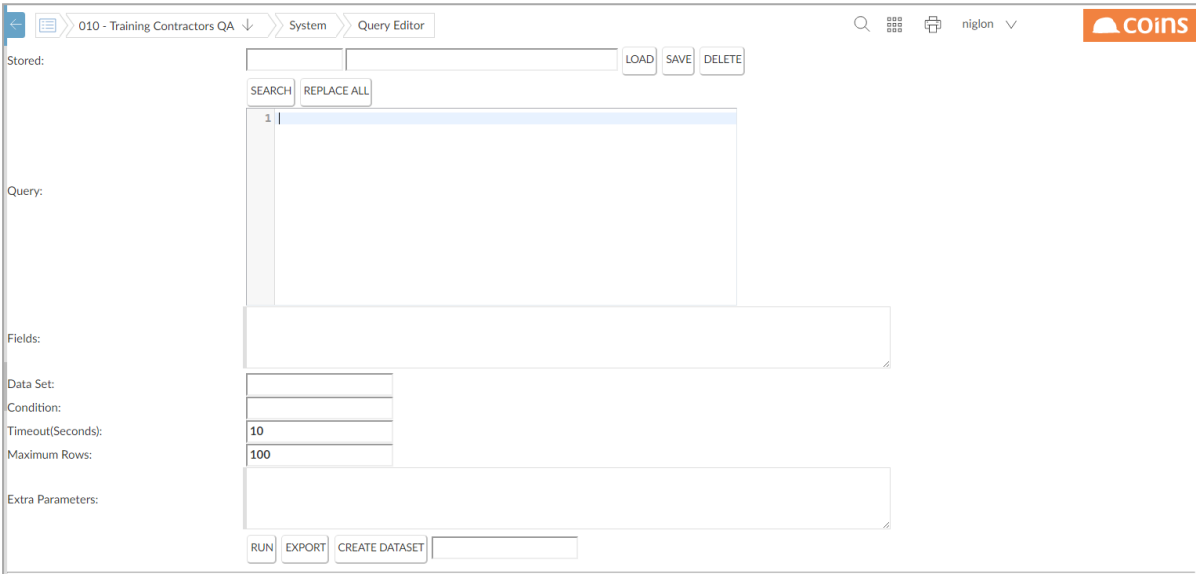
ADVANCED      Search Function      END

OPEN      NEXT

Access can then be assigned to Yes, No or Group

## 5.3 Query Editor

The Query Editor allows you to try out 4GL queries against the COINS database and sample the data returned. This function can be useful to test queries before being used in reports or enquiries.



The screenshot shows the COINS Query Editor interface. At the top, there is a breadcrumb trail: '010 - Training Contractors QA' > 'System' > 'Query Editor'. The interface includes a search bar with 'LOAD', 'SAVE', and 'DELETE' buttons. Below this is a 'Stored:' section with 'SEARCH' and 'REPLACE ALL' buttons. The main area is a large text editor for the query, with a line number '1' visible. Below the query editor are fields for 'Fields:', 'Data Set:', 'Condition:', 'Timeout(Seconds):' (set to 10), and 'Maximum Rows:' (set to 100). At the bottom, there is an 'Extra Parameters:' field and buttons for 'RUN', 'EXPORT', and 'CREATE DATASET'.

To use the query editor simply enter the query, and (optionally) any fields required - space separated - and click **RUN**. The system will return an error if any part of the query is incorrect - and a sample of data if the query compiles OK.

If the fields section was left blank, all fields will be displayed. If field names were specified, only those fields will be shown.



010 - Training Contractors QA System Query Editor

Stored: [ ] [LOAD] [SAVE] [DELETE]

Query: SEARCH [ ] REPLACE ALL [ ]

1 For EACH ap\_vendor WHERE kco = 10

Fields: [ ]

Data Set: [ ]

Condition: [ ]

TimeoutSeconds: 10

Maximum Rows: 100

Extra Parameters: [ ]

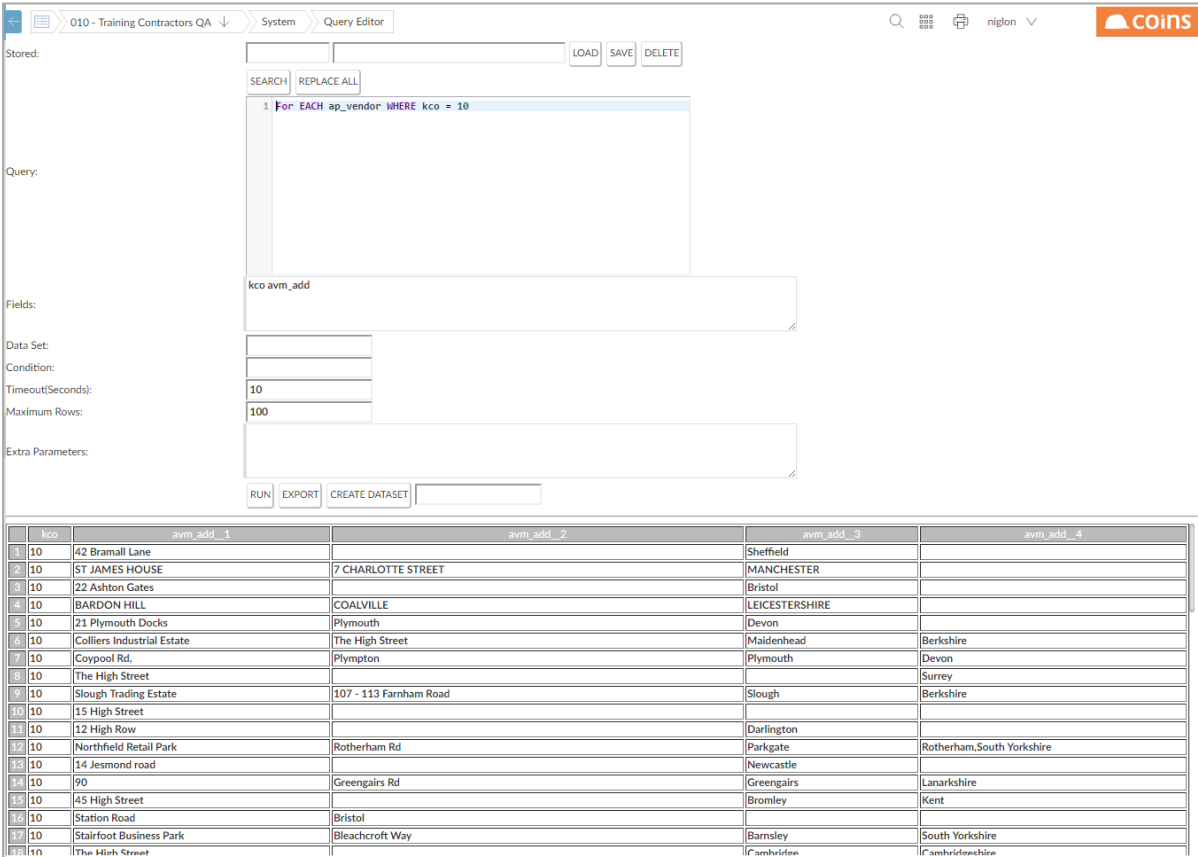
[RUN] [EXPORT] [CREATE DATASET] [ ]

id	amc_num	amc_name	amc_add_1	amc_add_2	amc_add_3	amc_add_4	amc_acode	amc_phone	amc_contact	amc_dctype	amc_ibcode	amc_disc	amc_datype	amc_daster	amc_hol	amc_fax	amc_factnum	amc_sclab	amc_slname	amc_srch
10	ABB001	Abbey Glass	42 Bramall Lane		Sheffield		S25 4DL	01642 897766		1	30	0.00	1	30	N	01642 897767		N	Abbey	Abbey Gla
10	ABSO004	Absolute Invoice Finance Limited	ST JAMES HOUSE	7 CHARLOTTE STREET	MANCHESTER		M1 4DZ			1	30	0.00	1	30	Y			N	ABSO	Absolute Invoice Finance Limited
10	AGG001	Aggregate Supplies	22 Ashton Gates		Bristol		BS30 5SJ	01675 556644		1	30	0.00	1	30	N	01675 556643		N	Aggregate	Aggregate Supplies
10	AGGR005	Aggregate Industries UK Ltd	BARDON HILL	COALVILLE	LEICESTERSHIRE		LE67 1TL	01530 511956		1	30	0.00	1	30	N	01530 815180		N	Aggregate	Aggregate Industries Ltd
10	AMS001	Amsterdam Imports Limited	21 Plymouth Docks	Plymouth	Devon		PL1 XYZ			1	30	0.00	1	30	N			N	Amsterdam	Amsterdam Imports Limited
10	APL001	A Plant Hire	Colliers Industrial Estate	The High Street	Maidenhead	Berkshire	SL6 3ND			1	30	0.00	1	30	N			Y	A Plant Hire	A Plant Hire
10	B&Q001	B&Q plc	Croypool Rd.	Plympton	Plymouth	Devon	PL7 4SS			1	30	0.00	1	30	N			N	B&Q Ltd	B&Q plc
10	BERTS001	Berts Bricks	The High							1	30	0.00	1	30	N			N	Berts Bricks &	Berts Brics

Field	Description
Data Set	A Data Set definition can be entered here to display the information created in the data set (No query or fields are required for this).
Condition Field	A function that determines whether a record should be included or not. The function returns a logical value: yes to include the record, no to exclude it.
Maximum Rows	Allows the query to run faster by only displaying a maximum number of rows per query. 10.23 onwards, this defaults to 10
Extra Parameters (10.23 onwards)	<p>Where a dataset has been specified, this field allows entry of parameters (URL) that are needed by the dataset query.</p> <p>e.g.</p> <p>The parameterised fields are so that you don't have to 'hard code' queries in the dataset to get it to run in the query editor – particularly if there are date replacements etc with fields like {RS_glp_fdate__2}. Or another useful reason for using these parameters is so that you can test results in a efficient way for instance: if you have a query on the dataset which reads:</p> <p>FOR EACH jc_job WHERE jc_job.kco = {kco} {jobSelect}</p> <p>You could call the dataset from within the query editor and in the parameters say jobSelect=and jc_job.job_num = 'XXXX' (where XXXX is a valid contract number).</p>

Field	Description
	<p>That way the dataset would run but for only contract XXXX - This is good to save time in checking the validation of fields in the dataset as you don't have to wait till the whole dataset evaluates prior to getting a response back.</p> <p>If you have more than one {} replacement in your dataset then you would separate the parameters with a &amp; symbol Eg: Dataset query might read :</p> <p>FOR EACH jc_job WHERE jc_job.kco = {kco} {jobSelect},</p> <p>EACH jc_costcode of jc_job WHERE TRUE {jccSelect}</p> <p>You could call the dataset from within the query editor and in the parameters say jobSelect=and jc_job.job_num = 'XXXX' &amp;jccSelect= and jc_costcode.jcc_cc = 'YYYY' (where XXXX is a valid contract number and YYYY is a valid Costcode).</p>

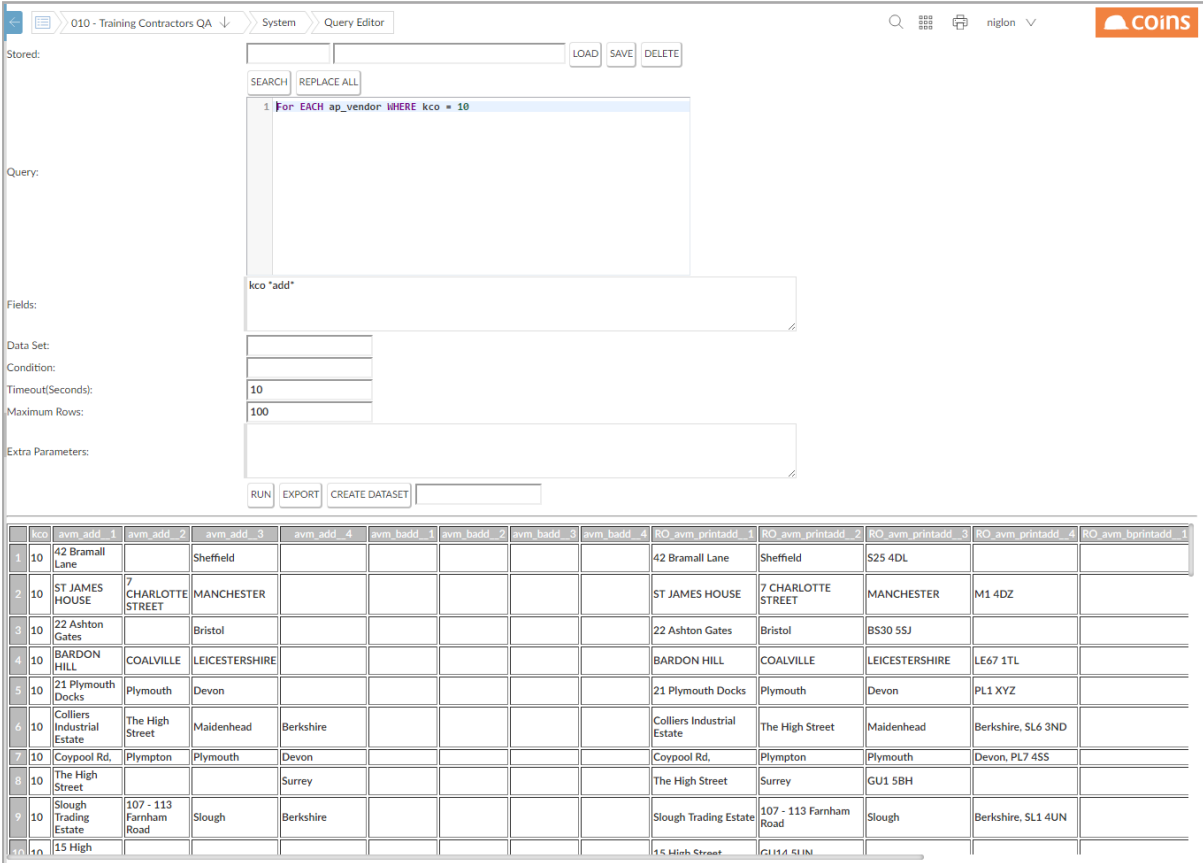
DB Fields defined as array in the Database Enquiry such as avm\_add[4] can be entered without the index of elements to display all elements (previously the element would need to be specified such as avm\_add\_\_1, avm\_add\_\_2 etc.)



The screenshot shows the COINS Query Editor interface. At the top, there's a navigation bar with '010 - Training Contractors QA', 'System', and 'Query Editor'. Below this, there are buttons for 'LOAD', 'SAVE', and 'DELETE'. The main area is divided into sections: 'Stored:', 'Query:', 'Fields:', 'Data Set:', 'Condition:', 'Timeout(Seconds):', 'Maximum Rows:', and 'Extra Parameters:'. The 'Query:' section contains the following SQL query: `1 For EACH ap_vendor WHERE kco = 10`. The 'Fields:' section shows `kco avm_add`. The 'Maximum Rows:' field is set to `100`. At the bottom, there are buttons for 'RUN', 'EXPORT', and 'CREATE DATASET'. Below the editor, a table displays the results of the query. The table has columns for 'kco', 'avm\_add\_\_1', 'avm\_add\_\_2', 'avm\_add\_\_3', and 'avm\_add\_\_4'. The data rows show various contractor details such as address, location, and county.

kco	avm_add__1	avm_add__2	avm_add__3	avm_add__4
10	42 Bramall Lane		Sheffield	
10	ST JAMES HOUSE	7 CHARLOTTE STREET	MANCHESTER	
10	22 Ashton Gates		Bristol	
10	BARDON HILL	COALVILLE	LEICESTERSHIRE	
10	21 Plymouth Docks	Plymouth	Devon	
10	Colliers Industrial Estate	The High Street	Maidenhead	Berkshire
10	Coypool Rd,	Plympton	Plymouth	Devon
10	The High Street			Surrey
10	Slough Trading Estate	107 - 113 Farnham Road	Slough	Berkshire
10	15 High Street			
10	12 High Row		Darlington	
10	Northfield Retail Park	Rotherham Rd	Parkgate	Rotherham, South Yorkshire
10	14 Jesmond road		Newcastle	
10	90	Greengairs Rd	Greengairs	Lanarkshire
10	45 High Street		Bromley	Kent
10	Station Road	Bristol		
10	Stairfoot Business Park	Bleachcroft Way	Barnsley	South Yorkshire
10	The High Street		Cambridge	Cambridgeshire

Wildcards can be used in “Fields” on Query Editor screen to mask fields




The screenshot shows the Query Editor interface with the following details:

- Stored:** [Empty field] [LOAD] [SAVE] [DELETE]
- Query:** `1 For EACH ap_vendor WHERE kco = 10`
- Fields:** `kco *add*`
- Data Set:** [Empty field]
- Condition:** [Empty field]
- Timeout(Seconds):** 10
- Maximum Rows:** 100
- Extra Parameters:** [Empty field]
- Buttons:** [RUN] [EXPORT] [CREATE DATASET] [Empty field]

kco	avm_add_1	avm_add_2	avm_add_3	avm_add_4	avm_badd_1	avm_badd_2	avm_badd_3	avm_badd_4	RO_avm_printadd_1	RO_avm_printadd_2	RO_avm_printadd_3	RO_avm_printadd_4	RO_avm_bprintadd_1
1	10	42 Bramall Lane		Sheffield					42 Bramall Lane	Sheffield	S25 4DL		
2	10	ST JAMES HOUSE	7 CHARLOTTE STREET	MANCHESTER					ST JAMES HOUSE	7 CHARLOTTE STREET	MANCHESTER	M1 4DZ	
3	10	22 Ashton Gates		Bristol					22 Ashton Gates	Bristol	BS30 5SJ		
4	10	BARDON HILL	COALVILLE	LEICESTERSHIRE					BARDON HILL	COALVILLE	LEICESTERSHIRE	LE67 1TL	
5	10	21 Plymouth Docks	Plymouth	Devon					21 Plymouth Docks	Plymouth	Devon	PL1 XYZ	
6	10	Colliers Industrial Estate	The High Street	Maldenhead	Berkshire				Colliers Industrial Estate	The High Street	Maldenhead	Berkshire, SL6 3ND	
7	10	Coypool Rd.	Plympton	Plymouth	Devon				Coypool Rd.	Plympton	Plymouth	Devon, PL7 4SS	
8	10	The High Street			Surrey				The High Street	Surrey	GU1 5BH		
9	10	Slough Trading Estate	107 - 113 Farnham Road	Slough	Berkshire				Slough Trading Estate	107 - 113 Farnham Road	Slough	Berkshire, SL1 4UN	
10	10	15 High							15 High Street	GL11A 5LN			

### 5.3.1 Search and Replace

Clicking **Search** allows you to search for a character string within the Query. This is particularly useful for large queries.



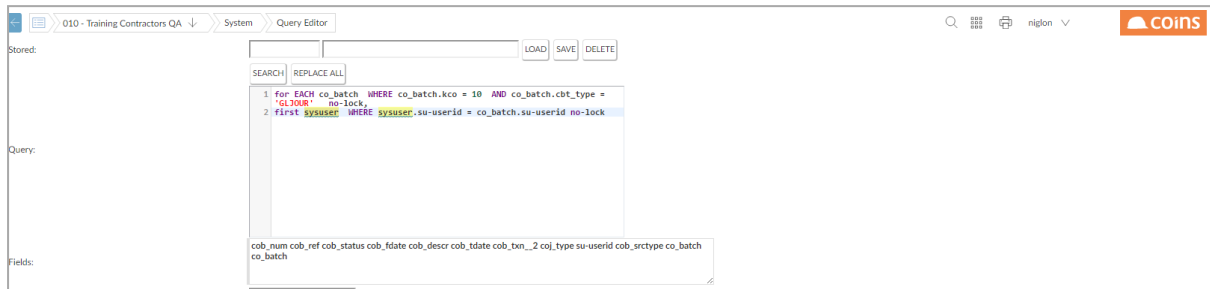
The screenshot shows the Query Editor interface with the following details:

- Stored:** [Empty field] [LOAD] [SAVE] [DELETE]
- Query:**

```
1 for EACH co_batch WHERE co_batch.kco = 'GLJOUR' no-lock,
2 first sysuser WHERE sysuser.su-userid
```
- Fields:** `cob_num cob_ref cob_status cob_fdate cob_descr cob_tdate cob_txn_2 coj_type su-userid cob_srctype co_batch co_batch`
- Search:** A search box is visible with the text "Search:" and a highlighted area in the query text.

If found, the string will be highlighted within the query.

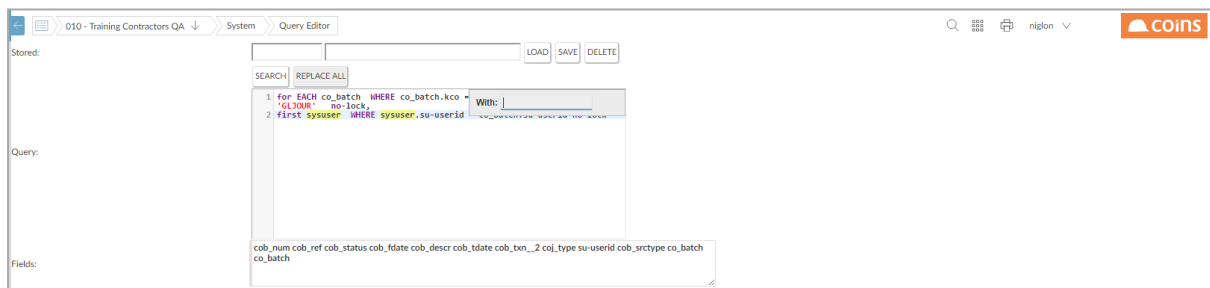




Clicking **Replace All** will prompt for the search string to be replaced



Enter the required string and click Enter. The system will then prompt for the replacement string.



Enter the replacement string and press Enter. The system will then replace all instances of original text with the replacement string.



### 5.3.2 Exporting from the Query Editor

To export the query and associated results to excel simply click **Export** . The query editor will then open the data set in a new screen. This information can then be exported to Excel by right-clicking anywhere in the data table.

The spread sheet created will contain the appropriate query and links to coins so that the data can be refreshed at any point.

### 5.3.3 Saving Queries in Query Editor (10.27)

To prevent the need to keep re-typing regularly used queries, save and load options are available as of v10.27 and are located at the top of the Query Editor page.

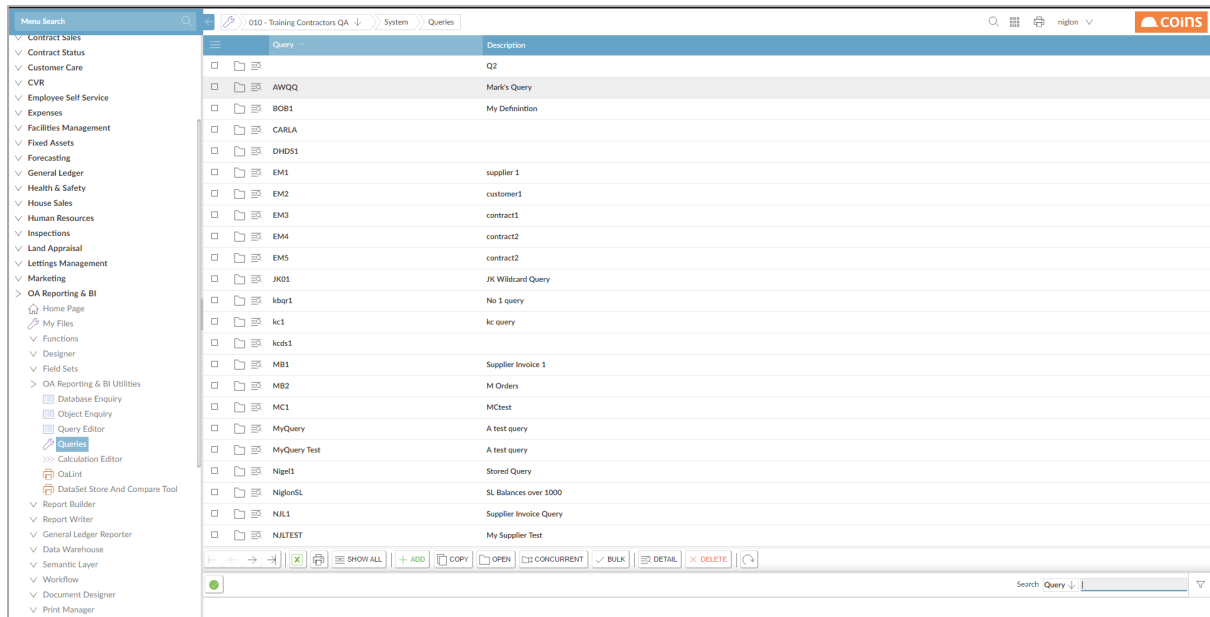
Once you have set up your query and selected the required fields, in the Stored fields, specify a name for the query and a description to further identify it.

Click **Save**

The Query will now be stored and can be retrieved at any time by running Query Editor, specifying the Query name and clicking **Load** button.

To delete a saved query, specify the name and select the Delete button.

The full list of stored queries may be viewed from the new Queries option from the OA Reporting and BI Utilities Menu



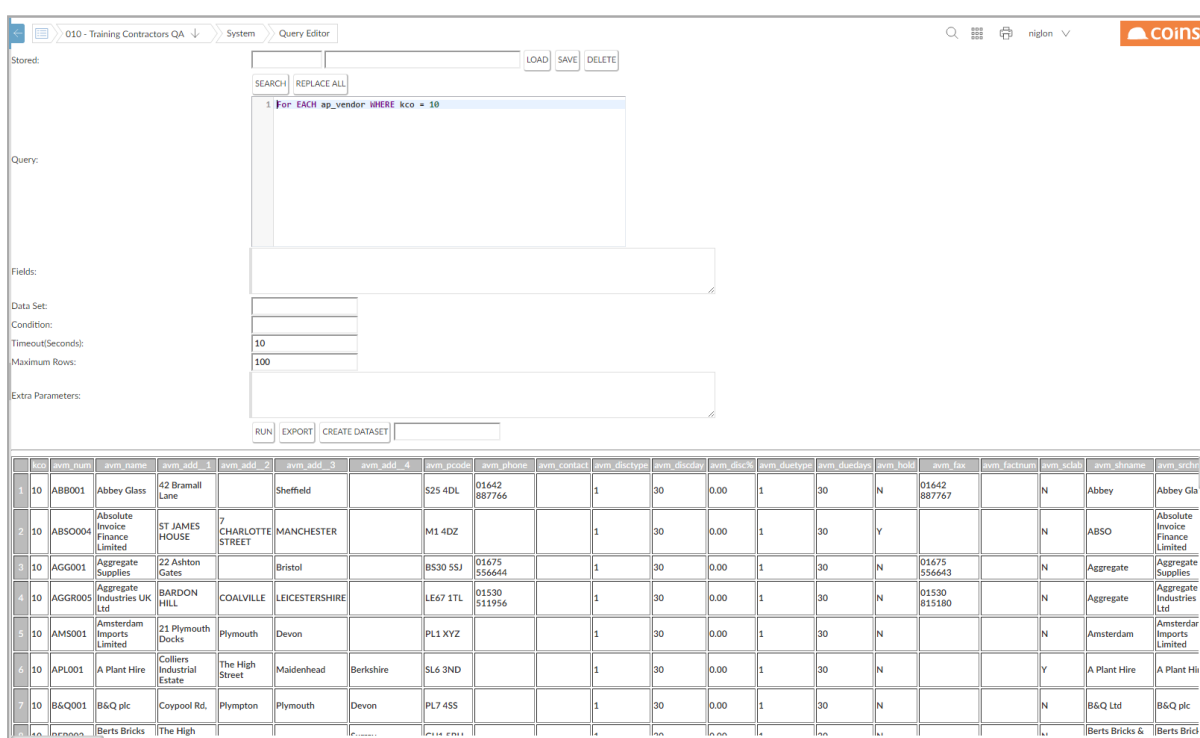
This screen will allow the creation, deletion and maintenance of the stored queries.

To see the results of the queries, you will need to return to Query Editor and Load the appropriate query.

### 5.3.4 Creating Datasets from the Query Editor

When building datasets, designers often use the Query Editor to test the query and select their fields before moving onto Dataset Maintenance and transferring the design to there. In 10.27 onwards, it is now possible to send the basic Dataset design directly into Dataset Maintenance without the need for manual re-keying.

A new button **Create Dataset** is now available at the bottom of the Query Editor fields, located next to the **Export** button:



kco	avm_num	avm_name	avm_add_1	avm_add_2	avm_add_3	avm_add_4	avm_postcode	avm_phone	avm_contact	avm_disttype	avm_distday	avm_distch	avm_disttype	avm_distdays	avm_hold	avm_tax	avm_factnum	avm_sclab	avm_slname	avm_srchl
10	ABB001	Abbey Glass	42 Bramall Lane			Sheffield	S25 4DL	01642 887766		1	30	0.00	1	30	N	01642 887767		N	Abbey	Abbey Gla
10	ABS0004	Absolute Invoice Finance Limited	ST JAMES HOUSE	7 CHARLOTTE STREET		MANCHESTER	M1 4DZ			1	30	0.00	1	30	Y			N	ABSO	Absolute Invoice Finance Limited
10	AGG001	Aggregate Supplies	22 Ashton Gates			Bristol	BS30 5SJ	01675 556644		1	30	0.00	1	30	N	01675 556643		N	Aggregate	Aggregate Supplies
10	AGG005	Aggregate Industries UK Ltd	BARDON HILL	COALVILLE		LEICESTERSHIRE	LE67 1TL	01530 511956		1	30	0.00	1	30	N	01530 815180		N	Aggregate	Aggregate Industries Ltd
10	AMS001	Amsterdam Imports Limited	21 Plymouth Docks	Plymouth		Devon	PL1 XYZ			1	30	0.00	1	30	N			N	Amsterdam	Amsterdam Imports Limited
10	APL001	A Plant Hire	Colliers Industrial Estate	The High Street		Maldenhead	Berkshire	SL6 3ND		1	30	0.00	1	30	N			Y	A Plant Hire	A Plant Hire
10	B&Q001	B&Q plc	Coypool Rd.	Plympton		Plymouth	Devon	PL7 4SS		1	30	0.00	1	30	N			N	B&Q Ltd	B&Q plc
10	BERTS001	Berts Bricks	The High							1	30	0.00	1	30	N			N	Berts Bricks &	Berts Bricks

Once you have a working query and field selection, enter a name for the new dataset and Select the **Create Dataset** button.

You need to specify the fields to be created in the dataset by adding them to the Fields section. Leaving this blank will result in the dataset fields section being empty.

The dataset will be created and will be available in Dataset Definitions in the OA Designer Menu.

## 5.4 Calculation Editor

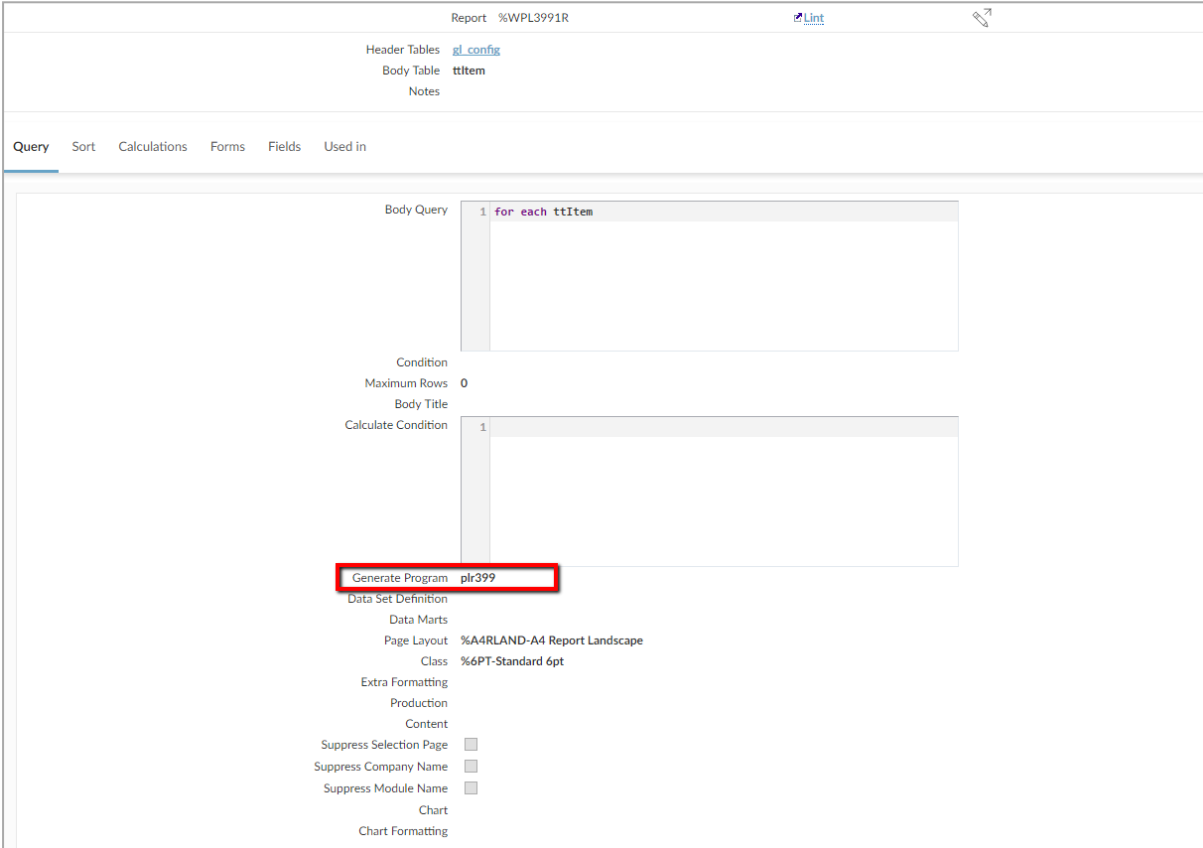
The Calculation Editor allows you to try out calculations using the coins business logic calculation methods (see accompanying documentation). This function can be useful to test calculations before being used in reports and enquiries.

Simply enter in the calculations using the appropriate syntax and click run to test the results.

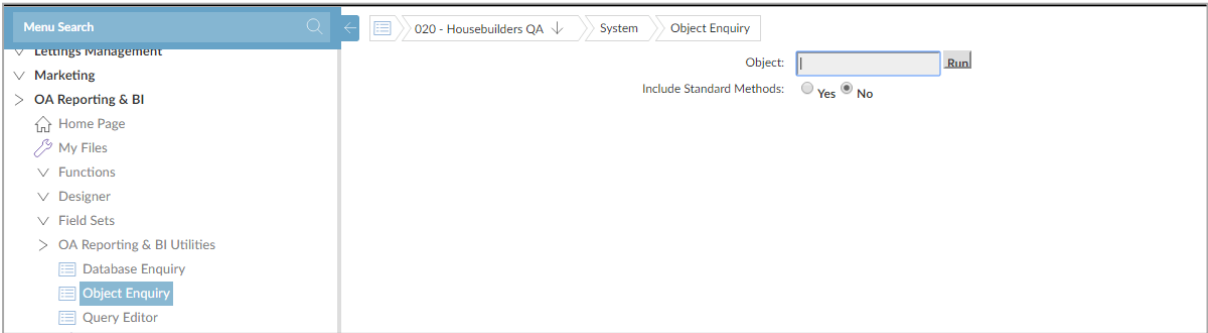
## 5.5 Object Enquiry

The object enquiry allows you to view the internal methods and reporting temporary tables (where present) for an object.

For example where coins may have written a standard report based upon a Generate Program such as the Purchase Ledger Open Item Report %WPL3991R – which queries a temporary table (ttlItem) generated by program plr399.



To enquire on the object, Select Object Enquiry in the OA Reporting and BI Utilities Menu.





In the Object field, enter the name of the object, for example: plr399 and select run (it is not necessary to include standard methods). At the bottom of the results, the object enquiry will return information on all of the reporting tables available within the generate program:

Menu Search

- Settings Management
- Marketing
- OA Reporting & BI
  - Home Page
  - My Files
  - Functions
  - Designer
  - Field Sets
  - OA Reporting & BI Utilities
    - Database Enquiry
    - Object Enquiry**
    - Query Editor
    - Queries
    - Calculation Editor
    - OaLint
    - DataSet Store And Compare Tool
  - Report Builder
  - Report Writer
  - General Ledger Reporter
  - Data Warehouse
  - Semantic Layer
  - Workflow
  - Document Designer
  - Print Manager
  - OA Reporting & BI Setup
  - Development Workspace

Object Name	Method	Input Fields
setTemplateFormat	PROCEDURE	
setTT	PROCEDURE	INPUT pcField CHARACTER, INPUT pcValue CHARACTER
startEval	PROCEDURE	
summaryRecord	PROCEDURE	INPUT pBuffer HANDLE, INPUT pcFields CHARACTER
summaryTT	PROCEDURE	INPUT pcFields CHARACTER
wou002-initialise	PROCEDURE	
writerRecord	PROCEDURE	

Reporting Tables

Table	Field	Data Type	Label	Documentation
ttItem	cEntity	character	Entity	Supplier
	rRowid	rowid	Rowid	Rowid(ap_invoice)
	cGan_code	character	Code	ap_invoice.gan_code
	cType	character	Type	Invoice = 'I' / Cheque = 'C'
	cAvn_num	character	Supplier	ap_invoice.avn_num
	cAin_inv	int Ref	ap_invoice.ain_inv	
	lacs_ref	integer	Pay Ref	ap_check.acs_ref
	clnt_Ref	character	clnt_Ref	If Invoice then ap_invoice.ain_inv else Cheque then ap_check.acs_ref
	cExt_Ref	character	cExt_Ref	ap_invoice.ain_supref
	clnvoice_Ref	character	clnvoice_Ref	When Invoice ap_invoice.ain_supref when Cheque then ap_check.acs_ref
	job_num	character	job_num	ap_invoice.job_num
	cAnal	character	cAnal	
	jDate	date	Inv date	If cByTm = Y and Invoice then ap_invoice.ain_idate else ap_invoice.ain_duedt
	jInvDate	date	Inv date	If Cheque then ap_check.acs_cdate
	jDueDate	date	Due date	
	dBalance	decimal	dBalance	If NOT asat report then ap_invoice.ain_balance else balance outstanding asat date.
	dCurrent	decimal	dCurrent	If the Balance of the Transaction < 0 then If the AgeCred = 'C' then the value of the Balance will be in the Current Field IF the Balance > 0 then the Value in Current will be Transactions Aged 0-29 Days
	dFuture	decimal	dFuture	If AgeCred='A' then ttItem.dcurrent = ?
	dFuture	decimal	dFuture	If Balance of Transaction > 0 and bUseFuture = TRUE AND Transaction Date >> Period Date of Report THEN ttItem.dfuture = dBalance.
	dUnalloc	decimal	dUnalloc	If the Balance of the Transaction < 0 then If the AgeCred = 'U' then the value of the Balance will be in the Unallocated Field.

## 6 End User Reporting

End User Reporting covers those elements of COINS BI that are intended to be used by general COINS users rather than the IT or Reporting Teams within a company.

These tools consist of:

- Report Writer
- Report Builder



## 6.1 OA Report Writer

The Report Writer is a tool which can be used by a broad cross-section of COINS users to produce powerful column based reports across the COINS system.

### 6.1.1 Key features

- Can be used 'out of the box', with over 1200 default queries, through the COINS Business Logic, providing access to every table in the system.
- Fully controllable using COINS function access so Users will only ever have [access](#) to the appropriate data.
- Suited to both managerial and operational end users of COINS.
- Users have ownership of their own reports, which can be published to COINS User Groups using the Report Runner functionality.
- Allows for quick and easy ad-hoc reporting across the COINS system.
- Supported by a documented view of the database ([Database Enquiry](#)).
- Can be a partner for, or a first step towards using the BI Designer toolset.

### 6.1.2 Key Functionality:

Whilst being an entry level reporting tool aimed at any COINS user, the Report Writer also provides extensive functionality to allow the production of powerful reports :

- Totalling methods.
- Dynamic sorting and summarising of the data.
- Using calculations to manipulate the data
- Displaying the information graphically using charts.



### 6.1.3 Granting Access to Report Writer and Report Runner

Report Writer licences are based on a Named Used licence. To assign a licence to a User, (once the appropriate module has been licensed via the branding screen) access the User record and click the appropriate licence.

Report Runner licences are based on a Named Used licence and can be assigned in the same manner as the Report Writer licence

### 6.1.3.1 Function Security

As well as having Report Writer licence ticked against their user profile, function security to the Report Writer will also need to be assigned to a user profile/groups:

The top level menu (for OA Reporting and BI) is %WOAREPBI

Below that they will need the OA Utilities/Database Enquiry:

%WOAREPU

%WSY5000WXXX

%WSY5001WXXX

Report writer consists of:

%WOAREP1

%WSY2400BRTN

%WSY2410BSTE

%WSY2400BRFD

%WSY2400BRFDA

%WSY2400BRFDB

%WSY2400BRFDD

%WSY2400BRFDU

%WSY2400BRFDX

%WSY2400BRTNA

%WSY2400BRTNB

%WSY2400BRTND

%WSY2400BRTNU

%WSY2400BRTNX

%WSY2400BXXX

%WSY2400BXXXX

%WSY2400FRTN

%WSY2400SRTN

%WSY2400SRTNO

%WSY2400SRTNT1

%WSY2400SRTNT2

%WSY2400SRTNT3

%WSY2400SRTNT4

%WSY2400SRTNT5

%WSY2400SRTNT6

%WSY2400SRTNT7

%WSY2400SRTNU

%WSY2401FRTN

%WSY2403FRTN

%WSY2405FRTN

#### **Report Runner**

For clients using Report Runner, in addition to the above, it will be necessary to grant the appropriate access to the Report Runner tab functions in the Report Status Workbench



(%WSY2430BRTN)

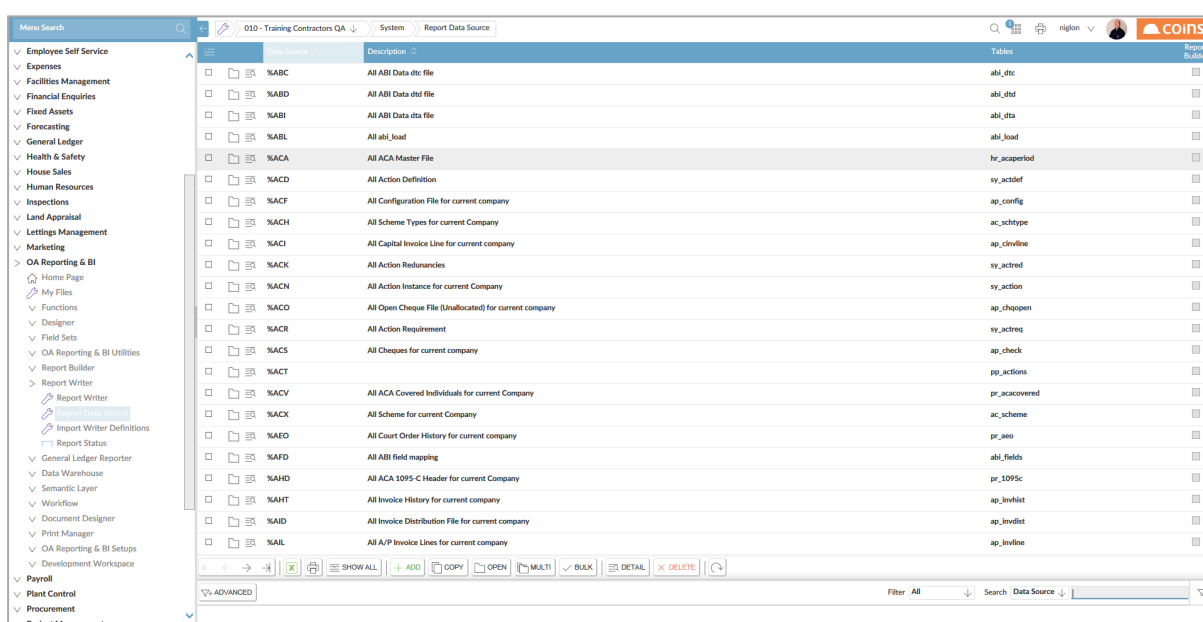
## 6.1.4 Report Data Sources

Report Data Source allows you to set up data sources for reports created using Report Writer or Report Builder. A large number of pre-defined queries exist to allow Report Writer to access the majority of the tables in COINS without the user having to know or use the database query language.

In addition to the standard set of Report Writer queries provided by COINS it is possible to for System Administrators, with a reasonable knowledge of the query language, to build user specific queries to meet business requirements not provided by the standard set.

The list of available Report Data Sources can be accessed by navigating to:

OA Reporting and BI > Report Writer > Report Data Source



Field	Description
Data Source	The code of the data source.
Description	The description of the data source.
Tables	Tables in the query
Report Builder	Whether this data source can be used in Report Builder

## 6.1.5 Report Data Source Definitions

Information from RDBMS systems is retrieved using query languages. Progress RDBMS (The database used by coins) uses Progress 4GL query language. In response to a query, the database returns a result set, which is just a list of rows containing the answers. The simplest query is just to return all the rows from a table, but more often, the rows are filtered in some way to return just the answer wanted.

Example Query on the coins database to retrieve all contracts (jc\_job) that belong to company 1:

```
FOR EACH jc_job WHERE jc_job .kco = 1
```

Often, data from multiple tables gets combined into one, by doing a join. Conceptually, this is done by taking all possible combinations of rows (the "cross-product"), and then filtering out everything except the answer.

Example Query on the coins database to retrieve all costheads (jc\_costcode) that belong to contracts (jc\_job) that belong to company 1:

```
FOR EACH jc_job WHERE jc_job .kco = 1,  
EACH jc_costcode OF jc_job
```

The flexibility of relational databases allows programmers to write queries that were not anticipated by the database designers. As a result, relational databases can be used by multiple applications in ways the original designers did not foresee, which is especially important for databases that might be used for decades. This has made the idea and implementation of relational databases very popular with businesses.

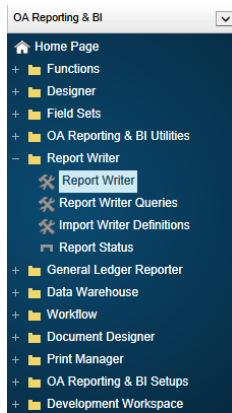
COINS OA uses a simplified version of the 4GL query language in combination with the RSP's to extract the data for reports and enquiries (for further information on RSP's – Record Service Procedures – see the relevant COINS BI documentation).

COINS OA uses the query to decide which records are accessed from the coins database from the database. The Page Design (Report Design) will determine which fields from these records are displayed (either on screen or in a report).

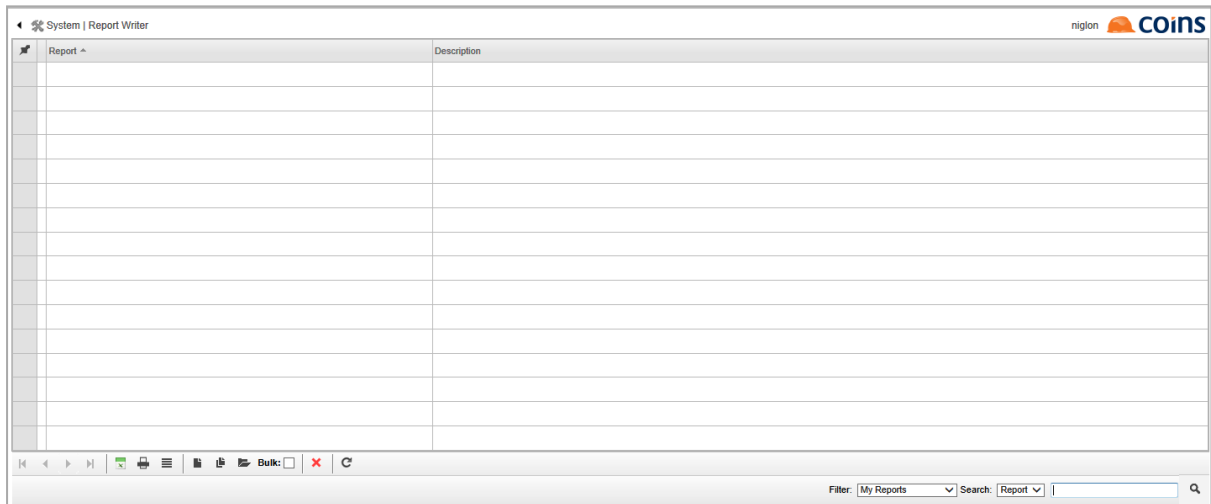
Field	Description
Notes	Notes or technical information about the query.
Query	4GL code that will select the appropriate records in a query.
Condition	<p>Query condition method for the query. The method specified is called and if the returned logical value is false then the record is excluded from the query.</p> <p>Conditions provide a means of giving additional functionality to the Query, but these must be pre-defined by COINS in order to be used and are stored in certain Record Service Procedures. If you have a requirement to access information in the data tables that does not appear to be straight-forward to put into standard Progress 4GL, please contact COINS for advice on possible Conditions that may be available.</p>
Calculate Condition	A calculate condition is used to limit the range of data returned to a report. The calculation is performed on each line of the report to determine whether the record should be included or not. The calculation should return either a 1 for true (i.e. display the line) or a 0 for false (do not display the line). The calculation should be a valid calculation on fields available to the report.
Input Form	<p>The input form used to prompt for run-time selection criteria for this report query. This is optional.</p> <p>It is possible to associate a Report Writer query with selections criteria, either standard or user defined. User Defined forms are created using the Page Designer Tool. The query must include the appropriate {tttSelect} token.</p>
Data Set Definition	<p>The data set definition(s) to be used for this query.</p> <p>A comma separated list of data set definition codes.</p>
Field List	<p>The list of fields that will be visible to the user of this query from the available fields in the query.</p> <p>You can enter a CAN-DO list of fields to be shown.</p>
Post Processing	




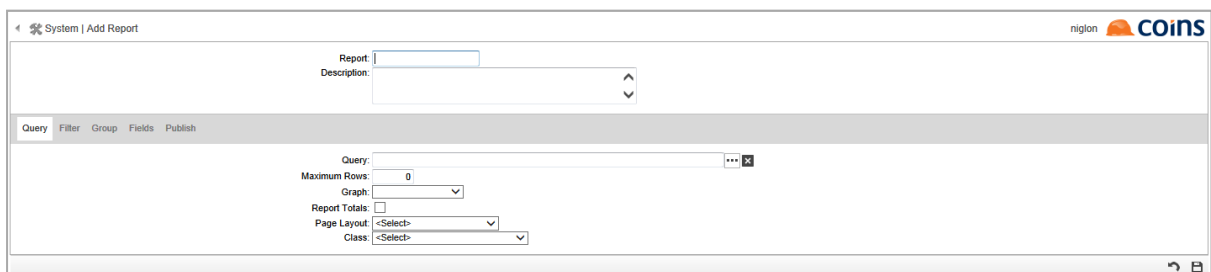
## 6.1.6 Report Writer - Creating a New Report



Report Writer is accessed from the OA Reporting and BI Menu.



To create a new report, select the New icon .



In this panel, assign a code and description to the report. The code will be used to identify the report later on and the description should contain a brief overview of the purpose of this report so that others Users will know what it does.

Report:

Description:

You will then need to select a Query from the pre-defined cache of queries.

Query:

Maximum Rows:

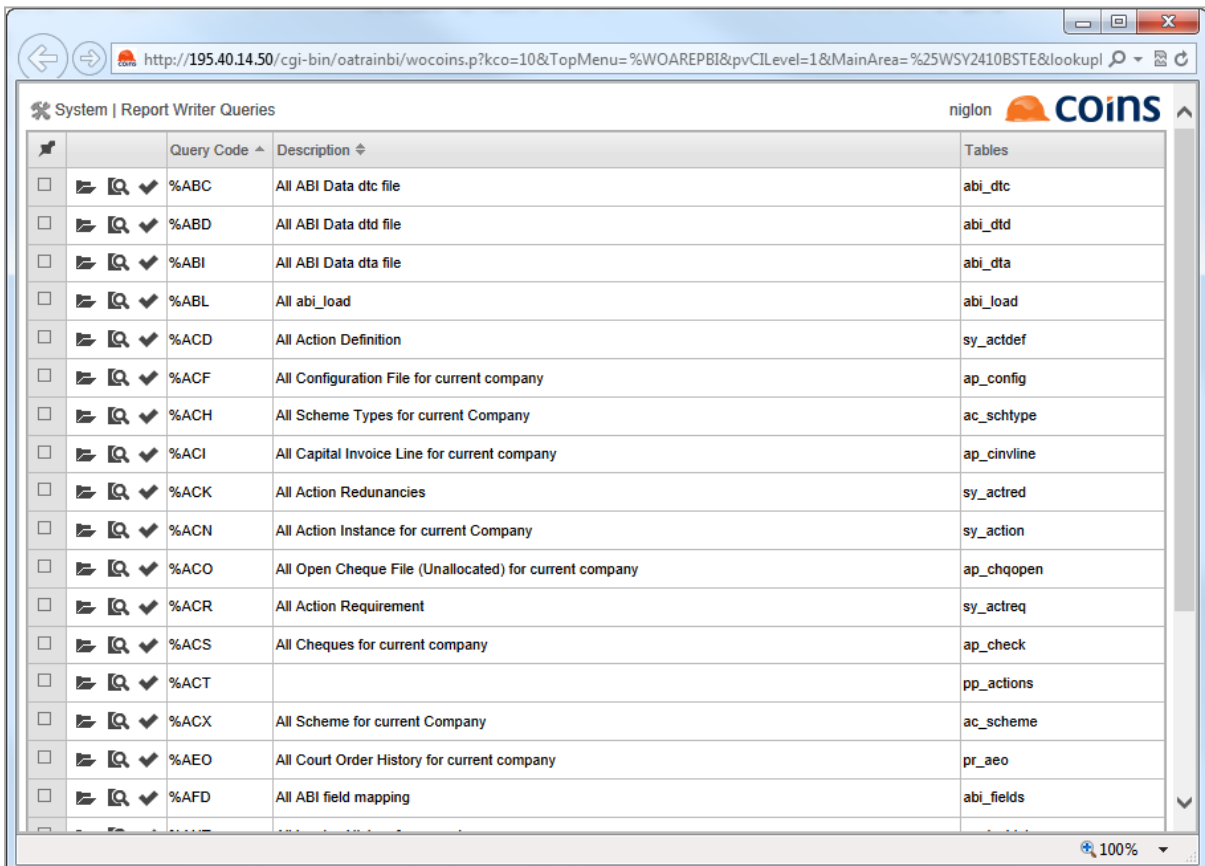
Graph:

Report Totals:

Page Layout:

Class:

Click on the Lookup button  to view all the available queries.

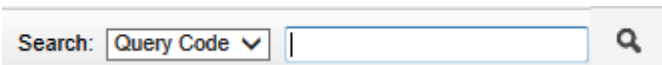


Query Code	Description	Tables
%ABC	All ABI Data dtc file	abi_dtc
%ABD	All ABI Data dtd file	abi_dtd
%ABI	All ABI Data dta file	abi_dta
%ABL	All abi_load	abi_load
%ACD	All Action Definition	sy_actdef
%ACF	All Configuration File for current company	ap_config
%ACH	All Scheme Types for current Company	ac_schtype
%ACI	All Capital Invoice Line for current company	ap_cinvline
%ACK	All Action Redunancies	sy_actred
%ACN	All Action Instance for current Company	sy_action
%ACO	All Open Cheque File (Unallocated) for current company	ap_chqopen
%ACR	All Action Requirement	sy_actreq
%ACS	All Cheques for current company	ap_check
%ACT		pp_actions
%ACX	All Scheme for current Company	ac_scheme
%AEO	All Court Order History for current company	pr_aeo
%AFD	All ABI field mapping	abi_fields


A Query is a pre-defined section of PROGRESS 4GL code that selects the records that the report will include. COINS comes with a large number of pre-installed queries and your system administrator may also have added some additional queries for your use.

NOTE : Queries are filtered by table security; you will only see queries for which you have access to all the tables.


To help you find the query you need, a filter is provided in the lower right-hand side of the screen

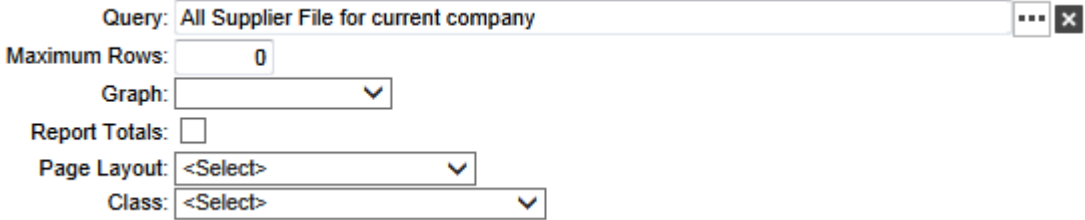


A search filter interface with a label 'Search:' followed by a dropdown menu set to 'Query Code', an empty text input field, and a magnifying glass icon.

You may search by Query Code, Query description, or the data tables used by the query. Remember to press the filter button  when you change the filter selection.

You can modify the basic queries later on by setting up filters, groups and sort orders.

The queries will extract the basic information for the report from the database and in some instances provide default Report Selection criteria. Select the query required and assign to the report using the Choose button .



A configuration form for a query. It includes a text field for the query name 'All Supplier File for current company', a 'Maximum Rows' field with the value '0', a 'Graph' dropdown menu, a 'Report Totals' checkbox, a 'Page Layout' dropdown menu with '<Select>' selected, and a 'Class' dropdown menu with '<Select>' selected.

If you want to report on a selected number of records then you can enter a number in the Maximum Rows field to limit the number of rows on the report. (Eg:- useful if you want to report on the Top 5). Leave this blank to run a full report.

If a graph is required on the report then select the graph type to be included on report.

---

Query: All Supplier File for current company ... X

Maximum Rows: 0

Graph: 2D Area  
2D Bar  
2D Pie  
3D Area  
3D Bar  
3D Horizontal Bar  
3D Pie  
Line

Report Totals:

Page Layout: 2D Pie ▼

Class: 3D Bar ▼

---

Using the Report Totals tick box, confirm whether you want the report to have run totals at the end of the report.

---

Query: All Supplier File for current company ... X

Maximum Rows: 0

Graph: 2D Area ▼

Report Totals:

Page Layout: <Select> ▼

Class: <Select> ▼

---

It is also necessary to confirm the Page Layout (this can be selected from a predefined list of standard options) and Class (again this can be selected from a predefined list of standard options).

---

Query: All Supplier File for current company ... X

Maximum Rows: 0

Graph: 2D Area ▼


Report Totals:

Page Layout: A4 Report Landscape ▼

Class: Standard 8pt (Standard) ▼

---

The Page Layout determines the paper size and orientation and the report class will determine font, size and formatting of the body text of the report.

Once you have assigned all of the above information Click  to save the report.

## 6.1.7 Selecting the fields to print on the report

### 6.1.7.1 New Reports

The fields Tab allows you to specify which fields from the data table will appear on the printed report.

When you are creating a new report you will be able to select the fields required from a list of all fields available list using the green arrows to select. Once a field has been moved to the selected column, it can be highlighted and removed again using the red arrows.

You can use the filter to find the fields you require. Simply type in a few letters of the field, or label required, in the box below the field names and the list will be dynamically filtered.

To change the order of the field in the selected column, highlight a field and use the up and down selectors to reposition the field in the list.

## 6.1.8 Sorting, Sub-totalling, Page Breaks and Summary Levels

The options under the Group Tab define how the data will be sorted on your report.

In the Group By column, select the field you want to group the report by. Records on which the value of this field are the same will be grouped together and sorted in the order you specify, and you can show totals for each group.

In the Label column, enter a label to show on the headers and footers (before and after each group).

NOTE : Your system administrator may set up more detailed group headers (using Default Report Labels), which show, for example, a code and a description for each group.

Choose whether to sort the group in ascending or descending order.

Choose whether or not to display group headers (a label and the value of the group by field) before each group, and group footers (a label and the value of the group field, plus the totals of any totalled columns).

If you want the report to start on a new page when the value of one of the fields changes (for example, to print information about different contracts on separate pages), use Page Level to select the field.

If you want to produce a summarised report based on details that you do not want to show, use Summary Level to select the field to summarise by.

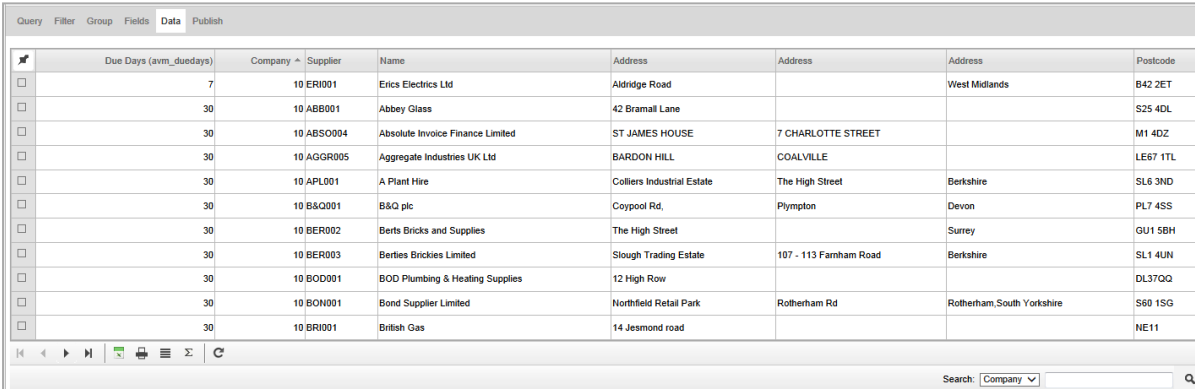
For example, if you want to produce a report on contract costs by section, you could produce a cost code report, but summarised by section. This would show the cost code totals for each section, but would not show the individual costs for each cost code.

Experiment with the options to see the diverse effects.


### 6.1.9 Previewing or exporting the data

When you are writing the report, the Data tab shows the data that the report will contain.

If the report requires selection criteria, the Input tab allows you to select which data to include, in the same way as when you run the report.



	Due Days (avm_duedays)	Company	Supplier	Name	Address	Address	Address	Postcode
<input type="checkbox"/>	7	10	ERI001	Eric's Electrics Ltd	Aldridge Road		West Midlands	B42 2ET
<input type="checkbox"/>	30	10	ABB001	Abbey Glass	42 Bramall Lane			S25 4DL
<input type="checkbox"/>	30	10	ABS0004	Absolute Invoice Finance Limited	ST JAMES HOUSE	7 CHARLOTTE STREET		M1 4DZ
<input type="checkbox"/>	30	10	AGGR005	Aggregate Industries UK Ltd	BARDON HILL	COALVILLE		LE67 1TL
<input type="checkbox"/>	30	10	APL001	A Plant Hire	Colliers Industrial Estate	The High Street	Berkshire	SL6 3ND
<input type="checkbox"/>	30	10	B&Q001	B&Q plc	Coypool Rd,	Plympton	Devon	PL7 4SS
<input type="checkbox"/>	30	10	BER002	Berts Bricks and Supplies	The High Street		Surrey	GU1 5BH
<input type="checkbox"/>	30	10	BER003	Berties Bricks Limited	Slough Trading Estate	107 - 113 Farnham Road	Berkshire	SL1 4UN
<input type="checkbox"/>	30	10	BOD001	BOD Plumbing & Heating Supplies	12 High Row			DL37 0Q
<input type="checkbox"/>	30	10	BON001	Bond Supplier Limited	Northfield Retail Park	Rotherham Rd	Rotherham, South Yorkshire	S60 1SG
<input type="checkbox"/>	30	10	BRI001	British Gas	14 Jesmond road			NE11

When you click the  [Apply Filter] button, COINS refreshes the Data tab with the data you specified.

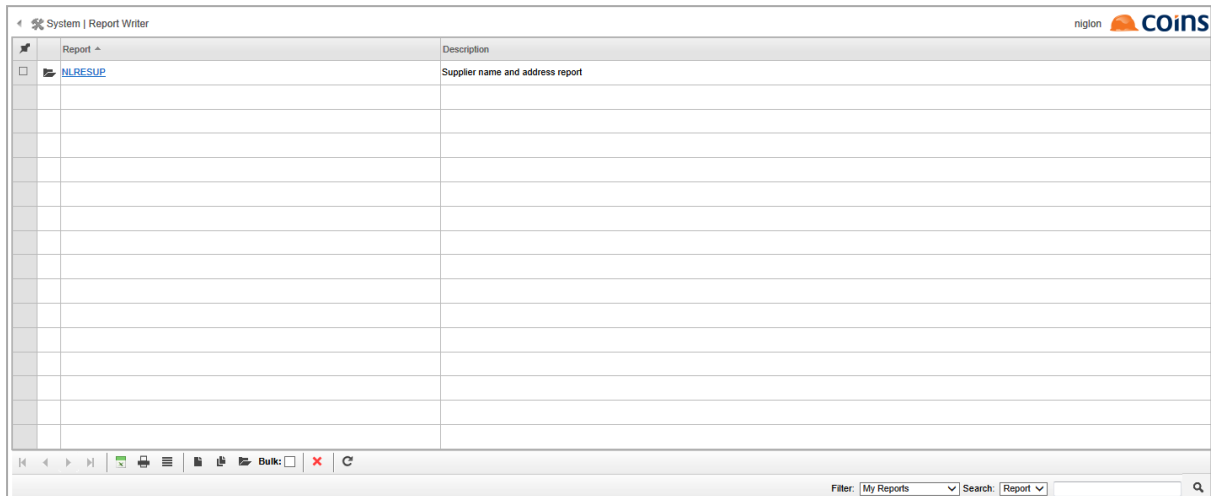
On the Data tab, you can also:

Show totals (for the fields marked to show totals): click the  button.

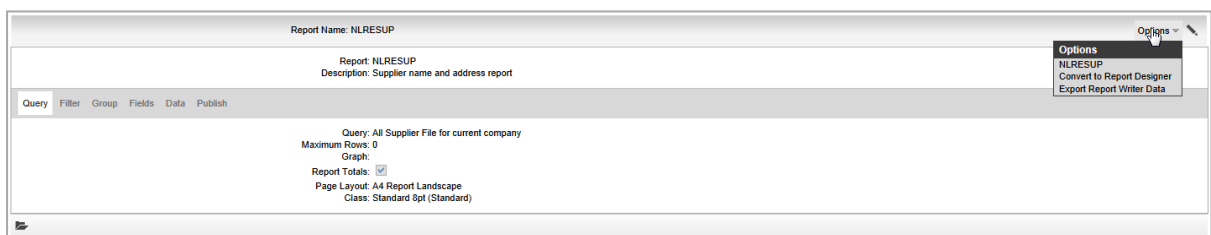
Export the data: click the  [Export] button.

### 6.1.10 Running a Report

1. Go to Report Writer.
2. Find the report you want to produce and click the link in the Report column.



3. From the Options menu, select the name of the report



4. Choose how you want COINS to produce the report:

- If the report requires selection criteria, you can select which data to include on the report. For example, if there is a contract selection tab, you can choose which contracts to report on, according to the different fields that are available on the tab.
- Choose the type of output you want. For example, you can choose to display the report on your screen as soon as it is ready (foreground). You can choose to email the report to someone, or to have it generated at a later date or time.

5. Click the forward button 

The report can then be accessed via the Report Status Workbench in the normal manner.

### 6.1.1 Publishing Report Writer Reports

Once a report has been created in Report Writer the author may wish to make this available to other COINS users who do not have access to Report Writer.

Three options are available:

- Publish to Report Runner – Requires a Report Runner Licence per user. This allows the Report Writer user to publish this report to the appropriate COINS User Groups. Those Users who have access both to the Report Runner and the defined User Group will have access to the run the Report via the Report Status Workbench.
- Export to Function – Creates a standard COINS function that may be added to any COINS menu by an Administrator. This requires no additional licencing.
- Export to Designer – Requires an OA Designer Licence. This is the same as Export to Function, but will open the report in OA Designer for additional Report Designer functionality to be added to the report.

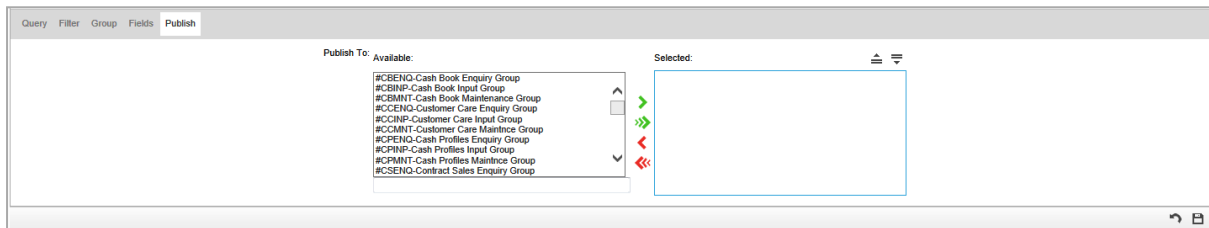


### 6.1.11. Publishing a report to Report Runner

To use this feature, you must have a Report Runner licence for each user who will need to access the report – they will only be able to run the reports, not make any changes to it.

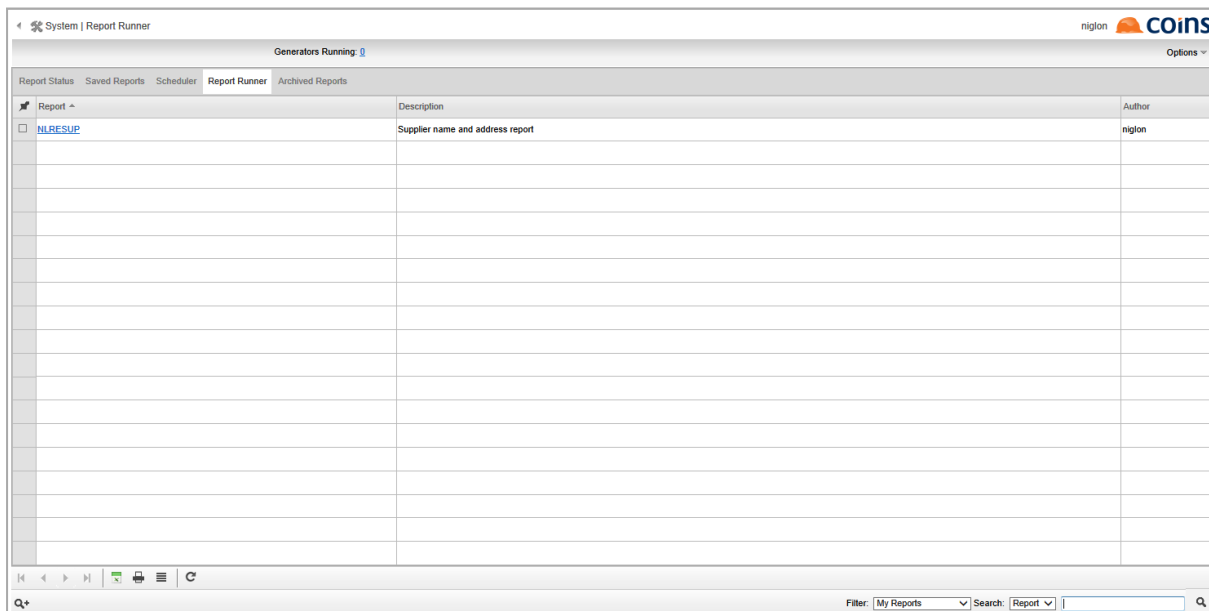
Open the report in Report Writer and select the Publish Tab.

Select the User Groups as required from the available list using the green and red arrows to select and deselect respectively.



You can use the filter to find the fields you require. Simply type in a few letters of the field or label required and the list will be dynamically filtered.

Those Users with access can then simply run the report from the Report Status Workbench by clicking on the link to the Report Code.



#### 6.1.11.1 Convert to Function

Open the desired report in Report Writer and click the Options button in the top right of the screen

From the drop-down menu, click Convert to Function

Field	Description
Report Code	Enter here the name of the function that is to be created. This may be the same name as the report in Report Writer (which may assist you in locating the report in the future if you need to make changes), or any other function name that meets your own internal standards. If in doubt, consult with your system administrator on a suitable naming convention. Do not include spaces or special characters in the function name.
Name	This is the description of the report that will appear on the COINS menus once your administrator has added it.  For example: Outstanding Invoices Report

Once you have completed the fields above, click

The function will be created and you will be returned to the Report Writer screen. You can now advise your system administrator of the details so that they can add this to a COINS menu and add the function to the required security groups (users will not be able to see the report unless they are granted security access to it).

If the function name has already been used, you will get a Function already exists error and you will either need to ask your Administrator to delete the old version (if appropriate), or choose a different Report Code).

#### Amending Reports that have been converted to Functions

Converted reports are separate to the Report Writer tool. If you need to make changes to a converted report, open the report in Report Writer and make any changes as required. You



will then need to ask your administrator to remove the existing function and Report Design before you can follow the steps above again.

### 6.1.11. Convert to Report Designer

If you have an OA Designer licence on your system, you can take a Report Writer Report and convert it to an OA Report Designer report. This will perform the same outcome as Convert to Function, but will then open the Report in OA Designer so that more complex report functionality can be added based on the original report design

Please note: Once you start to make changes in OA Designer, the report cannot then be maintained in Report Writer as OA Designer reports are not backwards compatible with Report Writer.

Open the desired report in Report Writer and click the Options button in the top right of the screen

From the drop-down menu, click Convert to Report Designer.

Field	Description
Report Code	Enter here the name of the function that is to be created. This may be the same name as the report in Report Writer (which may assist you in locating the report in the future if you need to make changes), or any other function name that meets your own internal standards. If in doubt, consult with your system administrator on a suitable naming convention. Do not include spaces or special characters in the function name.
Name	This is the description of the report that will appear on the COINS menus once your administrator has added it.  For example: Outstanding Invoices Report

Once you have completed the fields above, click

If the function name has already been used, you will get a Function already exists error and you will either need to ask your Administrator to delete the old version (if appropriate), or choose a different Report Code).

The function will be created and the report design will open in the OA Report Designer.

Once you have completed your design in OA Report Designer, you can advise your system administrator of the details so that they can add this to a COINS menu and add the function to the required security groups (users will not be able to see the report unless they are granted security access to it).

**Amending Reports that have been converted to OA Designer**

Converted reports are separate to the Report Writer tool. If you need to make changes to a converted report once you have made changes in OA Report Designer, you must continue to make any additional changes in Report Designer.

Reports in OA Designer are not backwards compatible with OA Report Writer.

### 6.1.12 Read Only (Calculated) Fields

In addition to the standard tables and fields in the COINS database, Open Architecture also provides access to certain calculated and non standard fields. These are known as “RO” or Read Only fields and are also fully documented in the Database Enquiry.<sup>1</sup>

RO_avm_factored_ext	Factor Information (More)	character	x(60)	RO	Text "Factored to" plus the factor account code, name, payment method and details.
RO_avm_gracctname	Name of Group Account	character	x(30)	RO	The name of the consolidating account.
RO_avm_hold_txt	Textual message for Hold	character	x(20)	RO	The name of the group account that is the group account for this supplier.
RO_avm_holdname	Name of Linked Account	character	x(30)	RO	"Supplier on hold" if the supplier is on hold (avm_hold).
RO_avm_linkname	Name of Linked Account	character	x(30)	RO	The name of the linked account.
RO_avm_method	Payment Method	character	X(1)	RO	The name of the linked account that is the linked account for this supplier.
RO_avm_method_non_factor	Extended payment method	character	x(60)	RO	Payment method and details for the supplier (if not paid by factor).
RO_avm_name*	Supplier Name (*)	character	x(40)	RO	Supplier name plus trailing * for printing on cheques.
RO_avm_OK	OK to Pay	decimal	->.>>.>>.>>.>>9.99	RO	The total of invoices that are due to be paid and are not held.
RO_avm_payeen	Payee Name	character	x(40)	RO	Payee name for the supplier.
RO_avm_payeen*	Payee Name (*)	character	x(40)	RO	Payee name for the supplier with trailing * for printing on cheques.
RO_avm_payterms	Payment Terms	character	x(20)	RO	Description of the payment due terms e.g. 30 days or 1 month due 28
RO_avm_sclab_details	Supplier Tax Details	character	x(60)	RO	If the supplier provides subcontract labour (avm_sclab=TRUE) then this is a composite description of the CIS details for the subcontractor.
RO_avm_title_age	Ageing: Title Age	character	x(10)	RO	Text "Age"
RO_avm_title_amt	Ageing: Title Amount	character	x(10)	RO	Text "Amount"
RO_avm_title_key	Ageing: Title Key	character	x(10)	RO	Text "Key"
RO_avm_title_qryamt	Ageing: Title Query Amount	character	x(12)	RO	Text "Query Amount"

Although these fields have certain restrictions, they are incredibly powerful when used in enquiries and reports. For example many of the calculated fields reflect similar fields to the COINS + Configurable Reporter, such as Accruals, Costs and Revenue fields.

There are three main types of calculated fields :

Simple that return descriptions from associated tables

Calculated that return calculated values from associated tables

Complex (also known as parameter drive) that return calculated values from associated tables based on parameter passed to the fields

---

1

#### 6.1.12.1 Using Parameter driven RO\_ fields in Report Writer

Certain RO\_ fields can be passed parameters to enhance the information returned to a report. Typically these fields can be limited by dates, values and financial periods as well simply parameters such as "TD" for a To date value.

The parameters available for each RO\_ field are documented in the Database Enquiry, below is an example :

In the Database Enquiry RO\_ fields are shown in a format as the example below. Any parameters immediately after the caret are mandatory, each parameter is then separated by a pipe. Any parameters which are encapsulated in square brackets are optional.

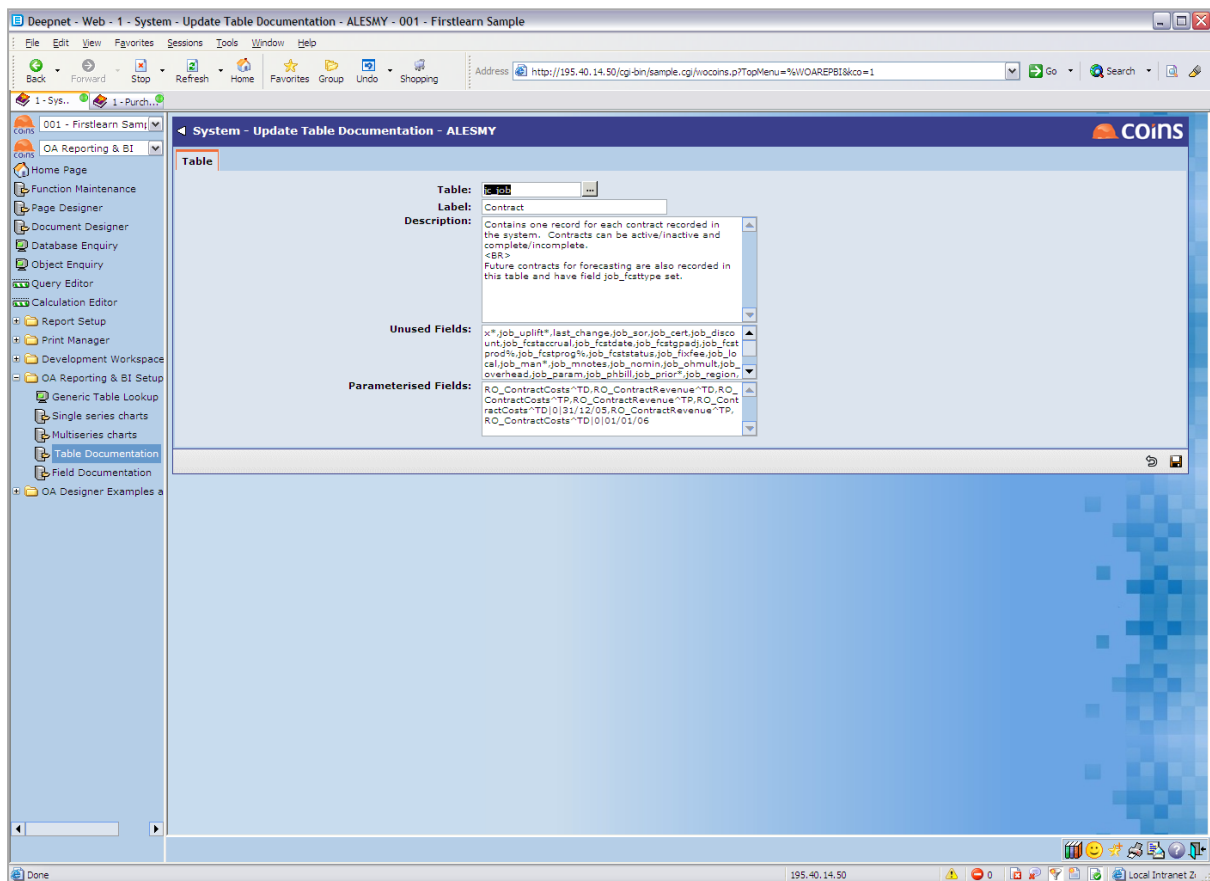
```
RO_ContractCosts^<PeriodType>[<PeriodOffset>[<FDate>[PhaseMasks[CostcodeMasks  
[CategoryMasks[AnalysisMask]]]]]
```

In order to use the Read Only (RO\_) fields in the Report Writer, they can either be exposed, i.e. document the parameters which are to be passed to a field, or they may be used directly in a calculation within a column on the report.

### 6.1.12. Exposing RO Fields

The benefit of using this method is the fields will be selectable by the user as report columns when building the report, however as RO Fields they will NOT be available for sorting and grouping. The disadvantage is that the fields will be hard-coded to the state they are exposed as and any change to their functionality must be made by the administrator and cannot be changed by the report writer users. For this reason, it is recommended that RO fields are used in calculations within the Report rather than by Exposing.

Go to the OA Reporting and BI Setup and Table Documentation and select the table in which the field you wish to pass parameters to.



The Parametrised Fields section then needs to be populated with the appropriate RO\_ field and any parameters that are required. (The parameter information can be found in the Database Enquiry).



A single field can be documented more than once with different parameters, for each a balance field can be documented with several dates, such as a year of period end dates.

Examples of an RO\_ field and associated parameters are :

**jc\_job.RO\_ContractCosts^TP|-3** (This Period with period offset -3)

**cs\_certificate.RO\_cst\_ctd\_cum^10** (Contract Sales Certificate Item – Cumulative to Date Certificate Line 10)

Once a RO\_ field has been entered in the Table Documentation with appropriate parameters it will then become available in Report Writer for selection as per standard database fields.

It will also appear in the Database Enquiry with a field type of EX



### 6.1.12.3 Using RO Fields directly in a calculation

RO fields, unless exposed, cannot be selected as a column by the user. However, they may be used within a calculation in a report column which allows the report writer user to directly control how the RO field is used and can change this at any time without needing to refer to a system administrator.

To create a calculated field, select ADD on the fields tab and leave the Field name blank. Enter a label for the field and within the calculation field enter the RO field and the required parameters. The field must be fully qualified as table.fieldname such as `jc_job.RO_ContractCosts^TD|0|`. If the table name is not specified, the field will not work as Report Writer will assume it is a variable rather than a database field.

Query Filter Group Fields Input Data Publish						
	Field	Variable	Label	Width	Total	Hidden
<input type="checkbox"/>	Contract (jc_job.job_num)		{jc Contract}	10mm	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	Name (jc_job.job_name)		Name	35mm	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>		dcosts	Costs	25mm	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	Total Types: Calculation: jc_job.RO_ContractCosts*TD Alignment: Default Format: Style:					

A variable name will need to be assigned in the Variable column if this column is to be included in Totals, as calculations require a variable to accumulate totals.

RO fields in calculations may also be passed values from sources such as input forms – if available in the query (for example period selections) using OA standard substitution {}. So if our input screen has a period field of `RS_glp_fdate__2`, our RO field might become:

`jc_job.RO_ContractCosts|0|{RS_glp_fdate__2}`. This RO field will then generate different figures depending on the period date selected by the user when they run the report, making the report much more dynamic.

Query Filter Group Fields Input Data Publish						
	Field	Variable	Label	Width	Total	Hidden
<input type="checkbox"/>	Contract (jc_job.job_num)		{jc Contract}	10mm	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	Name (jc_job.job_name)		Name	35mm	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>			Costs	30mm	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	Total Types: <input checked="" type="checkbox"/> Total Calculation: jc_job.RO_ContractCosts*TP 0 {RS_glp_fdate__2} Alignment: Default Format: Style:					
<input type="checkbox"/>			Costs -1	30mm	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>			Costs TD	30mm	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>			Costs TD -1	30mm	<input type="checkbox"/>	<input type="checkbox"/>

## 6.2 Semantic Layer Exercises - Overview

This section provides several exercises designed to help COINS administrator to setup the semantic layer for their end-users.

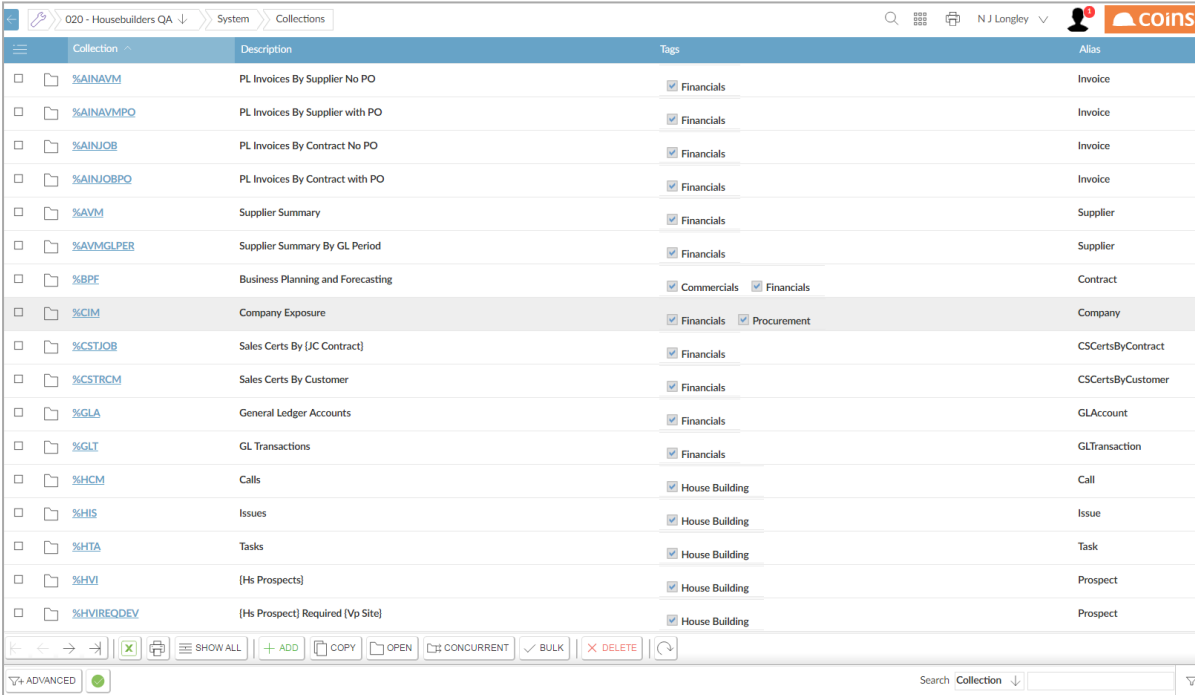
### 6.2.1 Exercise - A Simple Listing Report

In this exercise we are going to use the Semantic Layer to setup a Data Set for basic Contract Details and incorporate this into a report.

#### 6.2.1.1 Build the Data Set

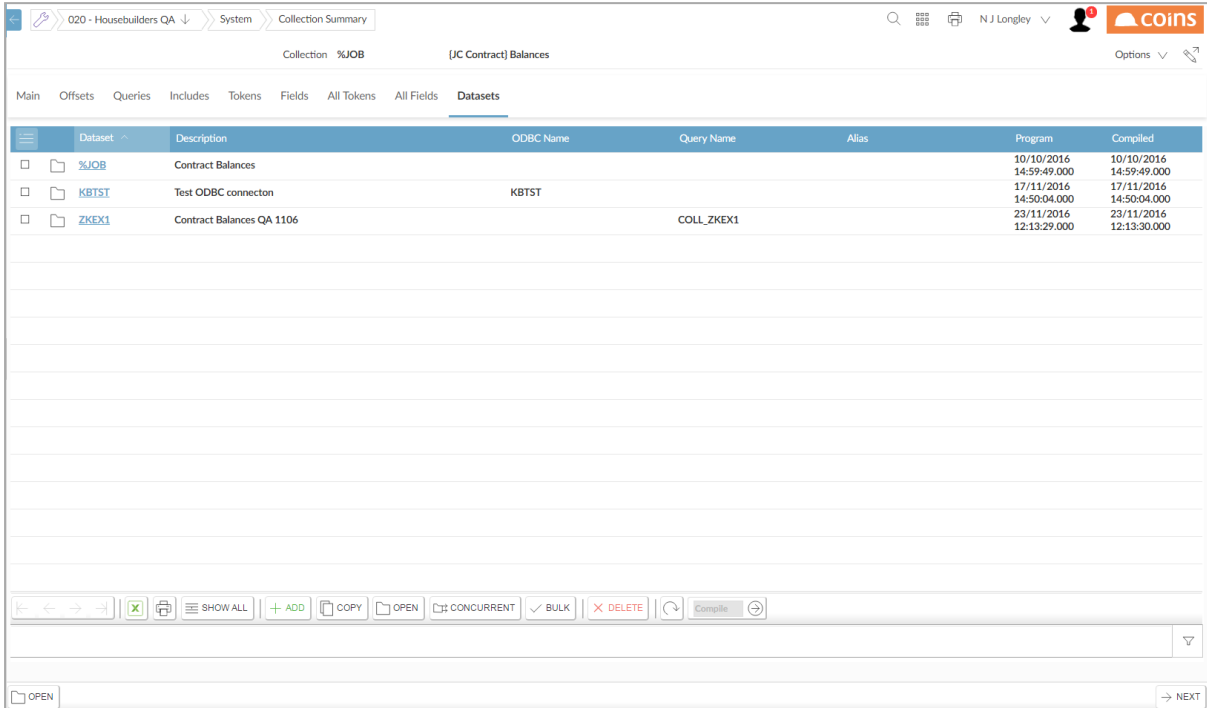
Navigate to the OA Reporting and BI module and select the Semantic Layer Sub-folder.

Select the Collections menu option.



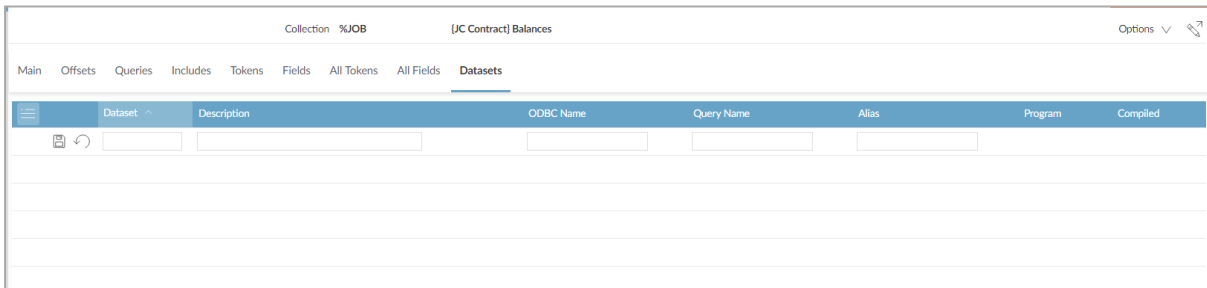
Collection	Description	Tags	Alias
<a href="#">%AINAVM</a>	PL Invoices By Supplier No PO	<input checked="" type="checkbox"/> Financials	Invoice
<a href="#">%AINAVMPO</a>	PL Invoices By Supplier with PO	<input checked="" type="checkbox"/> Financials	Invoice
<a href="#">%AINJOB</a>	PL Invoices By Contract No PO	<input checked="" type="checkbox"/> Financials	Invoice
<a href="#">%AINJOBPO</a>	PL Invoices By Contract with PO	<input checked="" type="checkbox"/> Financials	Invoice
<a href="#">%AVM</a>	Supplier Summary	<input checked="" type="checkbox"/> Financials	Supplier
<a href="#">%AVMGLPER</a>	Supplier Summary By GL Period	<input checked="" type="checkbox"/> Financials	Supplier
<a href="#">%BPF</a>	Business Planning and Forecasting	<input checked="" type="checkbox"/> Commercials <input checked="" type="checkbox"/> Financials	Contract
<a href="#">%CIM</a>	Company Exposure	<input checked="" type="checkbox"/> Financials <input checked="" type="checkbox"/> Procurement	Company
<a href="#">%CSTJOB</a>	Sales Certs By (JC Contract)	<input checked="" type="checkbox"/> Financials	CSCertsByContract
<a href="#">%CSTRCM</a>	Sales Certs By Customer	<input checked="" type="checkbox"/> Financials	CSCertsByCustomer
<a href="#">%GLA</a>	General Ledger Accounts	<input checked="" type="checkbox"/> Financials	GLAccount
<a href="#">%GLT</a>	GL Transactions	<input checked="" type="checkbox"/> Financials	GLTransaction
<a href="#">%HCM</a>	Calls	<input checked="" type="checkbox"/> House Building	Call
<a href="#">%HIS</a>	Issues	<input checked="" type="checkbox"/> House Building	Issue
<a href="#">%HTA</a>	Tasks	<input checked="" type="checkbox"/> House Building	Task
<a href="#">%HVI</a>	[Hs Prospects]	<input checked="" type="checkbox"/> House Building	Prospect
<a href="#">%HVIREQDEV</a>	[Hs Prospect] Required [Vp Site]	<input checked="" type="checkbox"/> House Building	Prospect

Select the Hyperlink on the %JOB Collection and select the Datasets Tab



Dataset	Description	ODBC Name	Query Name	Alias	Program	Compiled
%JOB	Contract Balances				10/10/2016 14:59:49.000	10/10/2016 14:59:49.000
KBTST	Test ODBC connecton	KBTST			17/11/2016 14:50:04.000	17/11/2016 14:50:04.000
ZKEX1	Contract Balances QA 1106		COLL_ZKEX1		23/11/2016 12:13:29.000	23/11/2016 12:13:30.000

Click



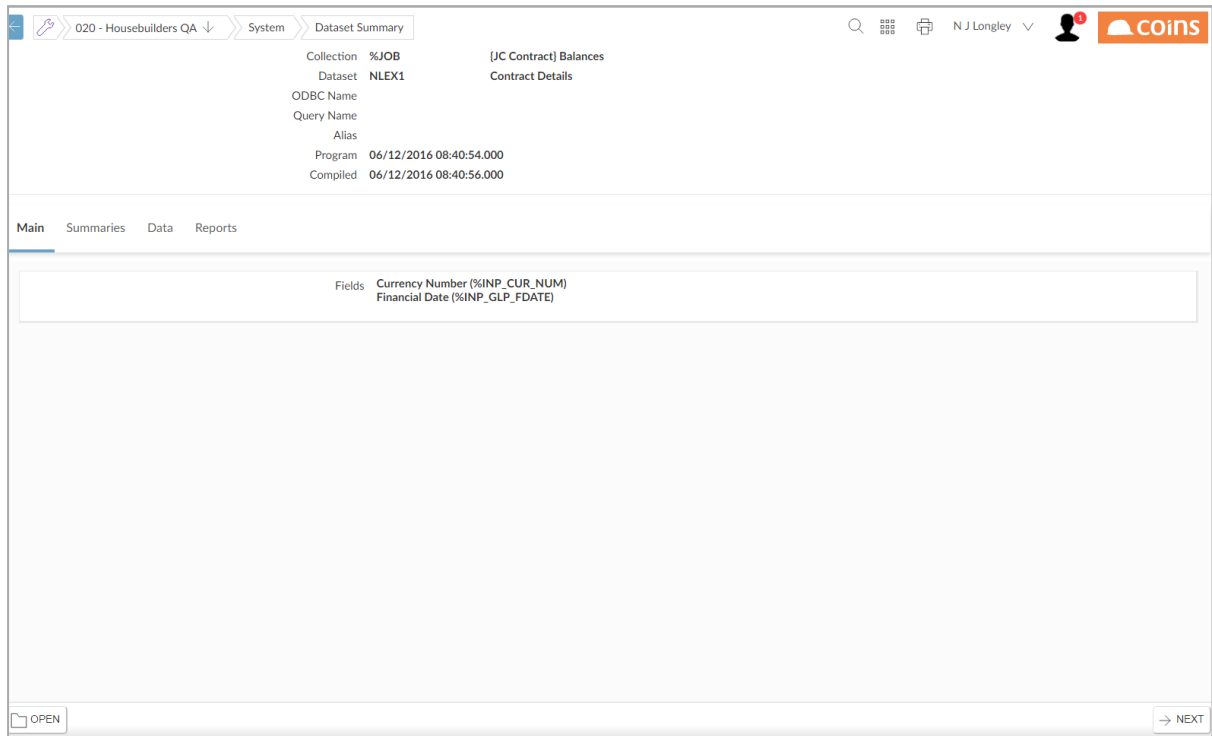
Dataset	Description	ODBC Name	Query Name	Alias	Program	Compiled
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>		

Enter a Dataset name of xxEX1 (Where xx are your initials).

Enter a description of Contract Details

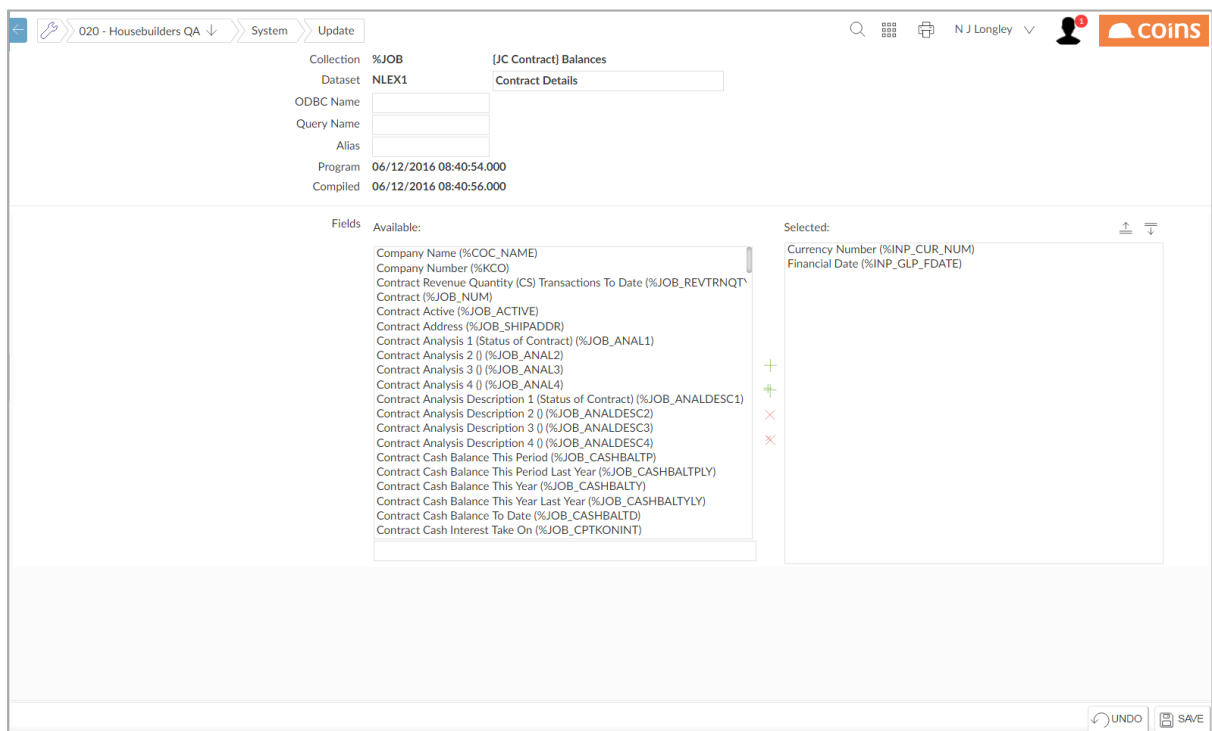
Leave ODBC and Query Name details blank at this stage – we will return to these later

Click



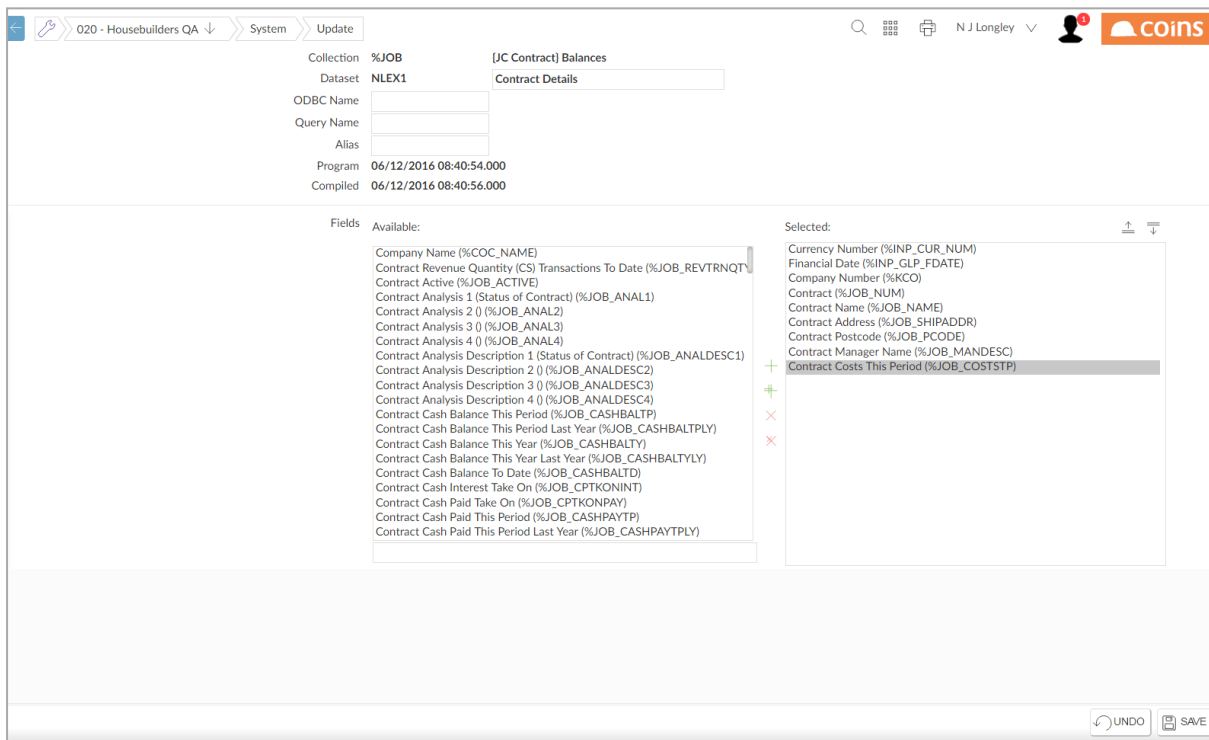
The Main Tab will display any mandatory fields that are required for the operation of the dataset.

Click



From the Available field locate the following fields and move then to the Selected field.

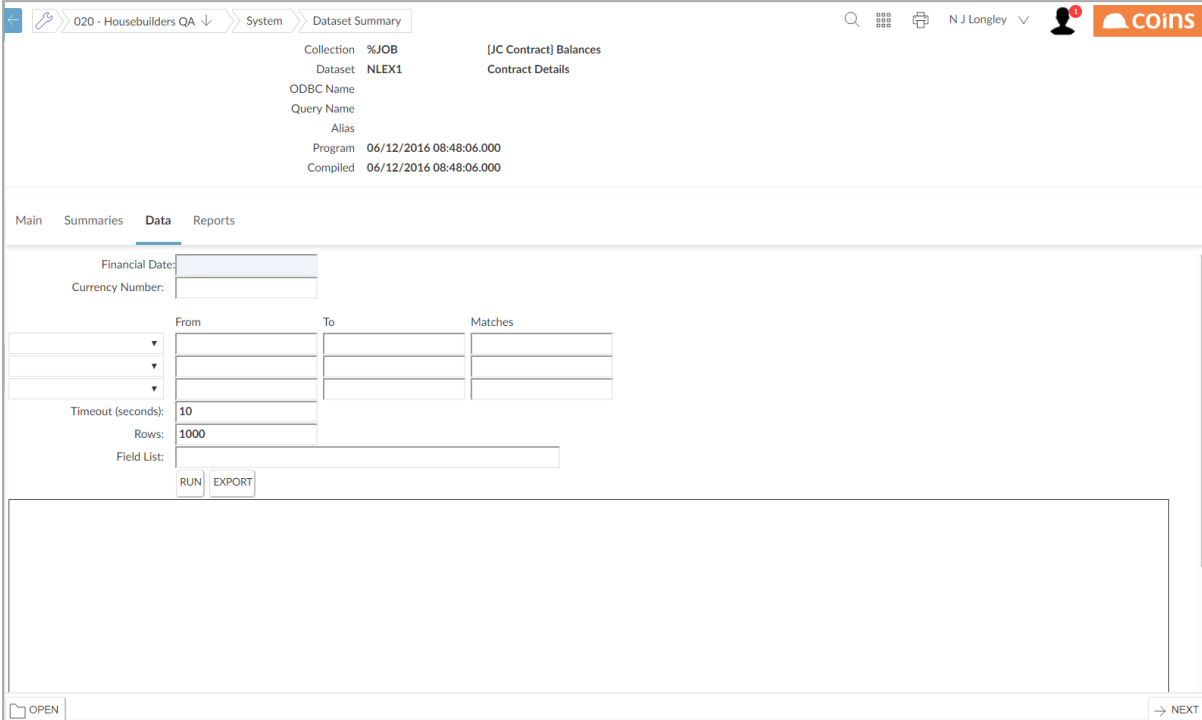
- Company Number (%KCO)
- Contract (%JOB\_NUM)
- Contract Name (%JOB\_NAME)
- Contract Address (%JOB\_SHIPADDR)
- Contract Post Code (%JOB\_PCODE)
- Contact Manager Name (%JOB\_MANDESC)
- Contract Costs This Period (%JOB\_COSTSTP)



Click

6.2.1.2 Test the Dataset

Select the Data tab



020 - Housebuilders QA >> System >> Dataset Summary

Collection %JOB [JC Contract] Balances  
Dataset NLEX1 Contract Details  
ODBC Name  
Query Name  
Alias  
Program 06/12/2016 08:48:06.000  
Compiled 06/12/2016 08:48:06.000

Main Summaries **Data** Reports

Financial Date:   
Currency Number:

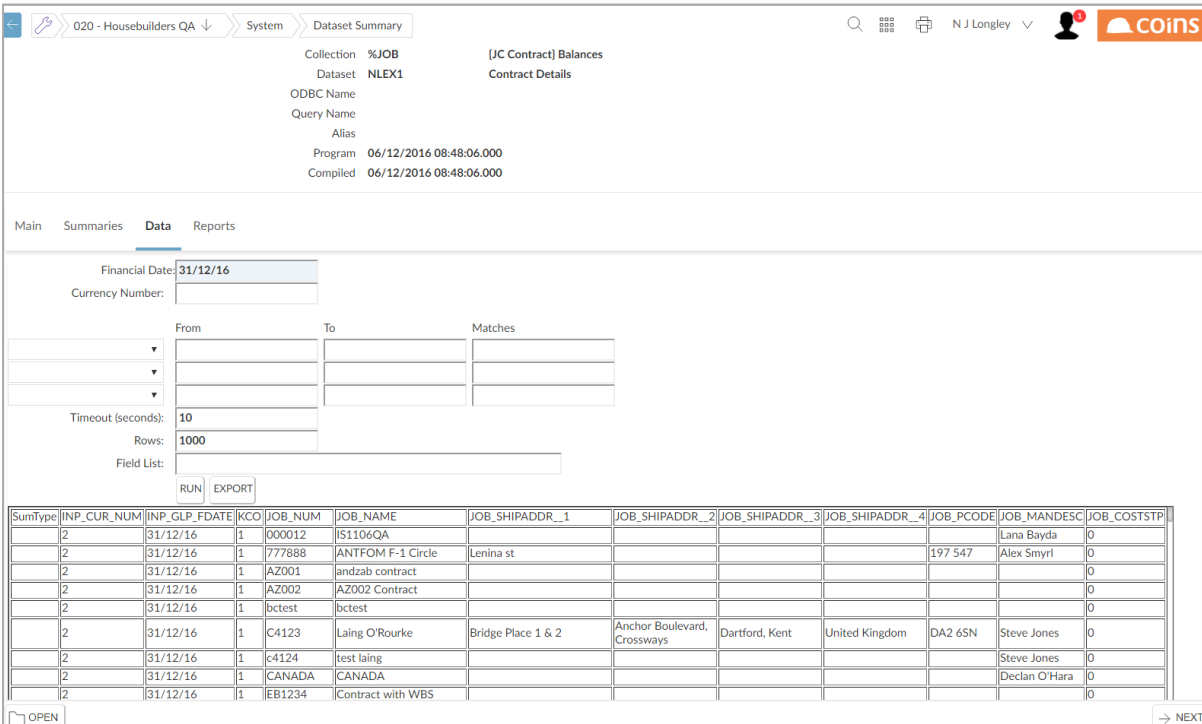
From To Matches

Timeout (seconds): 10  
Rows: 1000  
Field List:

RUN EXPORT

OPEN NEXT

The yellow field indicates where a mandatory value is required – in this case it is the Financial Period on which we want to report. Enter a valid Financial Date and click **Run**



020 - Housebuilders QA >> System >> Dataset Summary

Collection %JOB [JC Contract] Balances  
Dataset NLEX1 Contract Details  
ODBC Name  
Query Name  
Alias  
Program 06/12/2016 08:48:06.000  
Compiled 06/12/2016 08:48:06.000

Main Summaries **Data** Reports

Financial Date:   
Currency Number:

From To Matches

Timeout (seconds): 10  
Rows: 1000  
Field List:

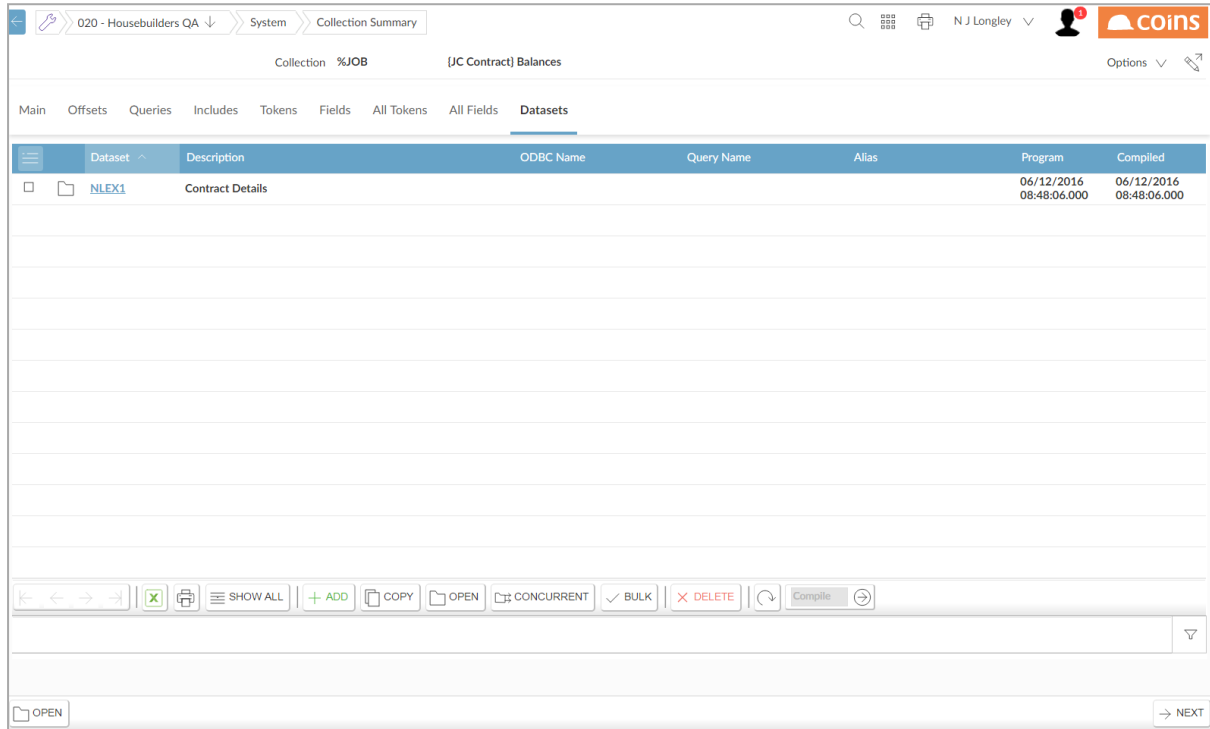
RUN EXPORT

SumType	INP_CUR_NUM	INP_GLP_FDATE	KCO	JOB_NUM	JOB_NAME	JOB_SHIPADDR_1	JOB_SHIPADDR_2	JOB_SHIPADDR_3	JOB_SHIPADDR_4	JOB_PCODE	JOB_MANDESC	JOB_COSTSTP
2		31/12/16	1	000012	IS1106QA						Lana Bayda	0
2		31/12/16	1	777888	ANTFOM F-1 Circle	Lenina st				197 547	Alex Smyrl	0
2		31/12/16	1	AZ001	andzab contract							0
2		31/12/16	1	AZ002	AZ002 Contract							0
2		31/12/16	1	bctest	bctest							0
2		31/12/16	1	C4123	Laing O'Rourke	Bridge Place 1 & 2	Anchor Boulevard, Crossways	Dartford, Kent	United Kingdom	DA2 6SN	Steve Jones	0
2		31/12/16	1	c4124	test laing						Steve Jones	0
2		31/12/16	1	CANADA	CANADA						Declan O'Hara	0
2		31/12/16	1	EB1234	Contract with WBS							0

OPEN NEXT

### 6.2.1.3 Make the Dataset available to the Reporting Tools

Return to the Collection %JOB and select the Dataset Tab



Open the Dataset by clicking  against the Dataset Name.





The screenshot shows the 'Datasets' tab in the COINS BI Toolset. The table has the following columns: Dataset, Description, ODBC Name, Query Name, Alias, Program, and Compiled. The first row contains the following data:

Dataset	Description	ODBC Name	Query Name	Alias	Program	Compiled
NLEX1	Contract Details				06/12/2016 08:48:06.000	06/12/2016 08:48:06.000

In the Query Name field, enter COLL\_xxEX1 (Where xx are your initials). We recommend prefixing Semantic Layer Query Names with COLL\_ as it makes locating them easier later on.

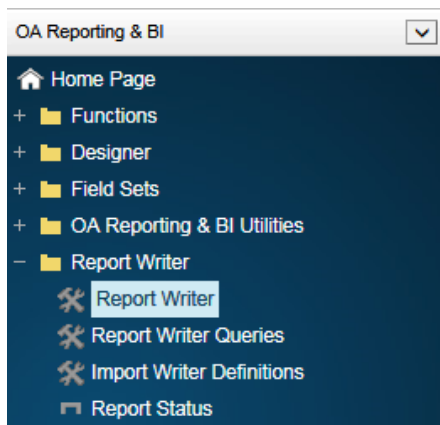
The screenshot shows the 'Datasets' tab in the COINS BI Toolset. The table has the following columns: Dataset, Description, ODBC Name, Query Name, Alias, Program, and Compiled. The first row contains the following data:

Dataset	Description	ODBC Name	Query Name	Alias	Program	Compiled
NLEX1	Contract Details		COLL_NLEX1		06/12/2016 08:48:06.000	06/12/2016 08:48:06.000

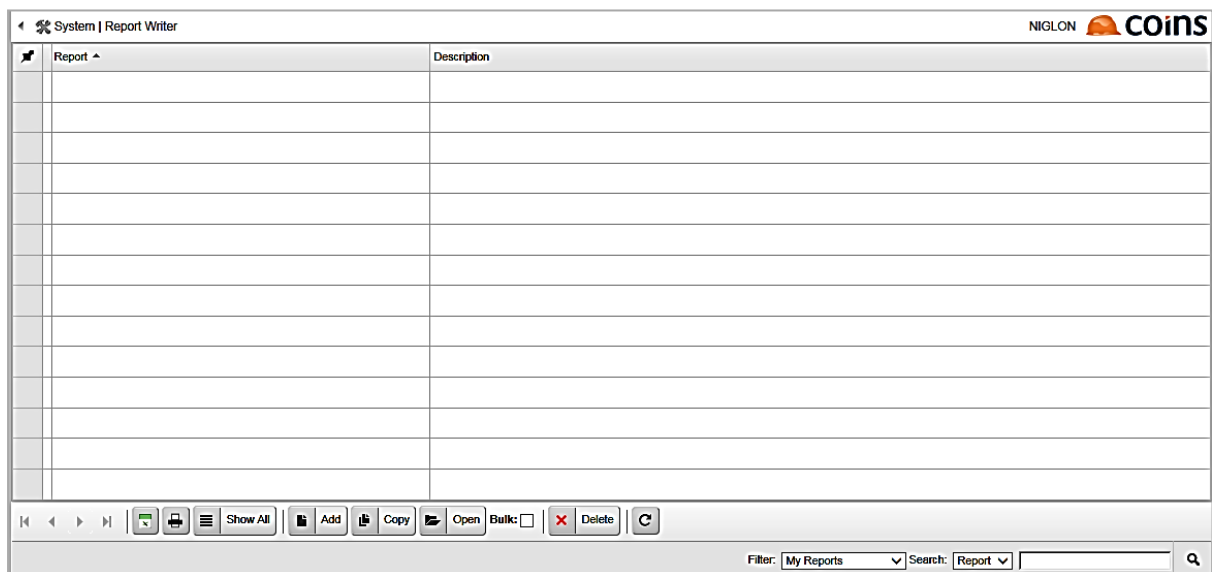
Click

## 6.2.2 Using the Dataset in a Report Writer Report

This exercise assumes familiarity with the Report Writer Tool and therefore does not go into detail on its operation. A separate detailed guide on Report Writer is available from the COINS Learning Resources section in the Client Area on the COINS Website.



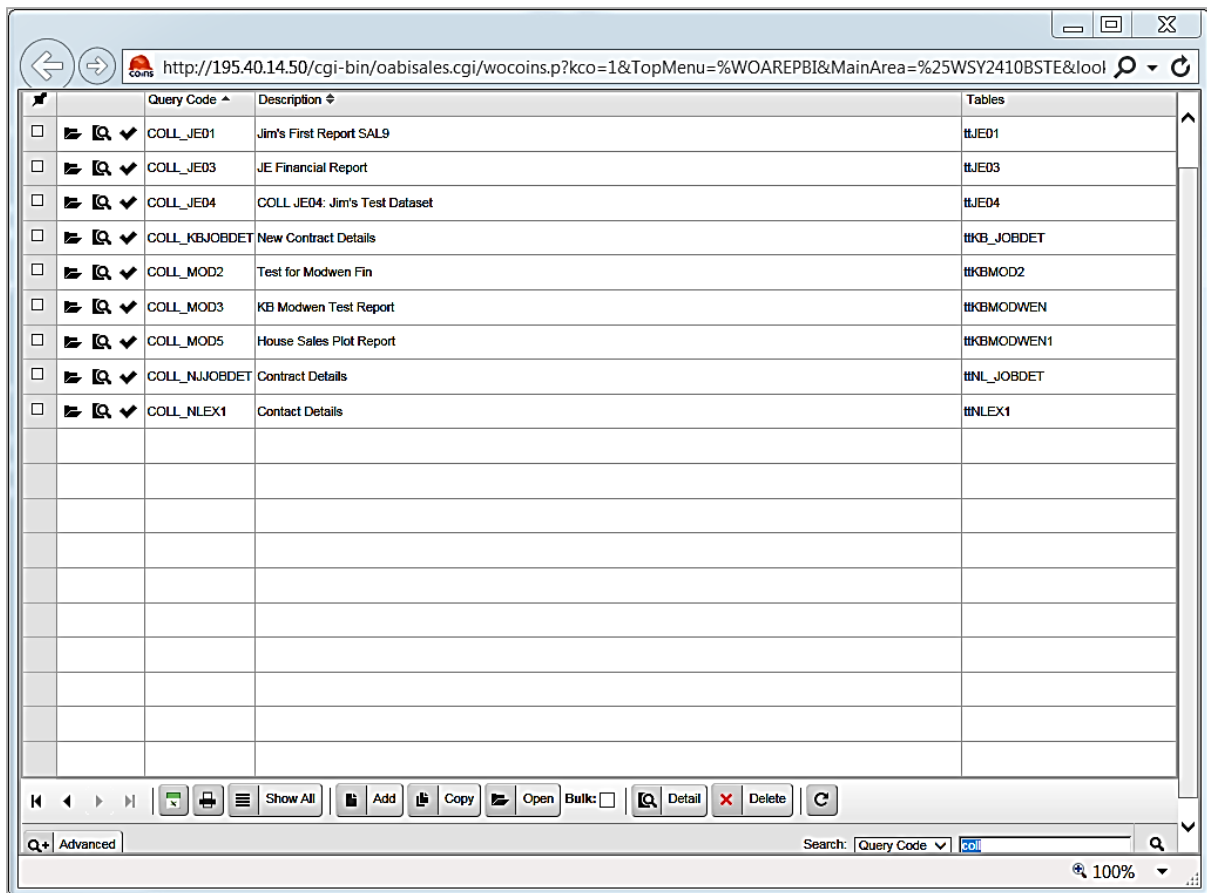
In the OA Reporting and BI Module, select the Report Writer Sub folder and select the Report Writer option



Click 

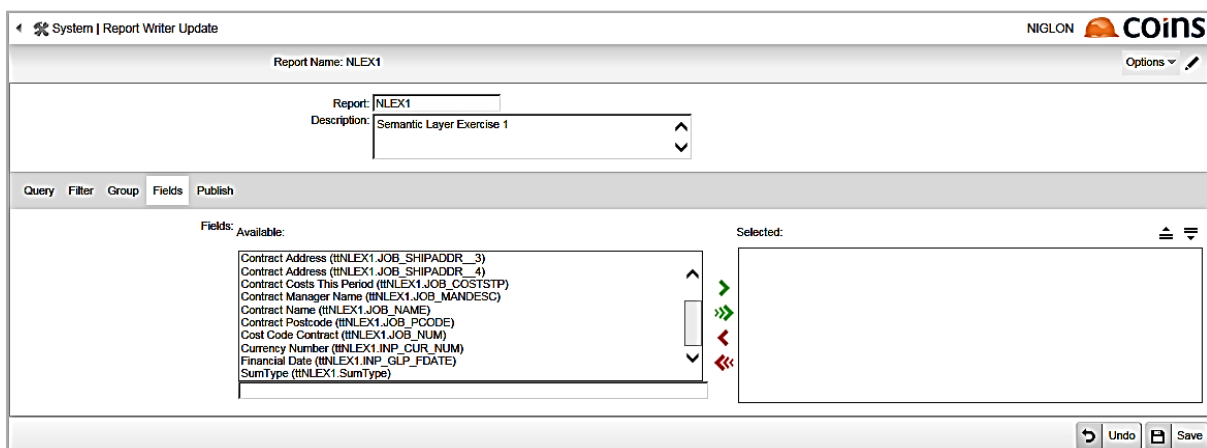
Give your report a name and description and click the Query lookup .


Set the Search filter to Query Code and COLL



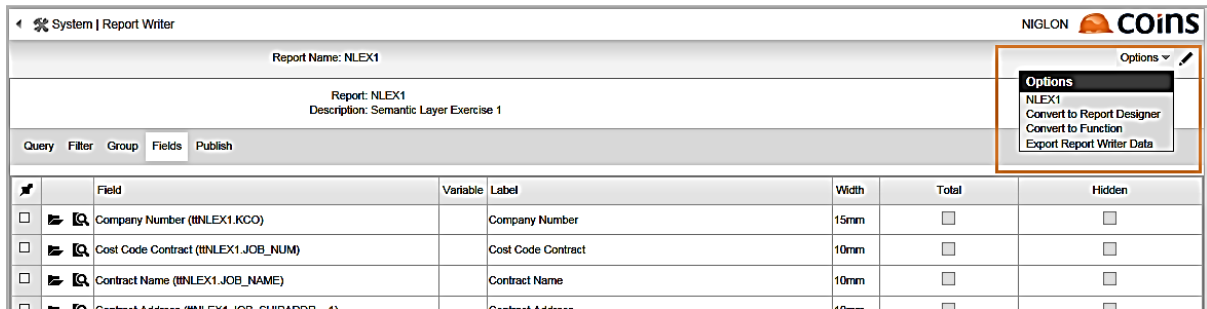
Select the Query COLL\_xxEX1 (Where xx is your initials) using the Select  button. This Query was created when you specified the Query Name against your Dataset within the Semantic Layer.

The fields available for selection are those as defined within the Dataset

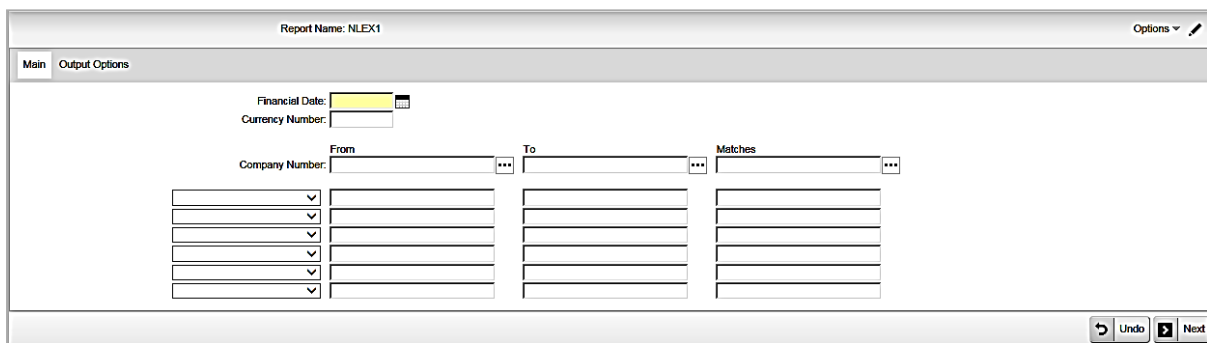


Select your fields arrange the order as required. Click .

To run the report, click Options in the upper right of the screen



Select the Report name from the drop down menu



The selection screen from the Data tab on the Dataset definition is offered to the user for the record selection criteria. If any additional filtering is required, this can be built into the Report design using the Filter Tab.

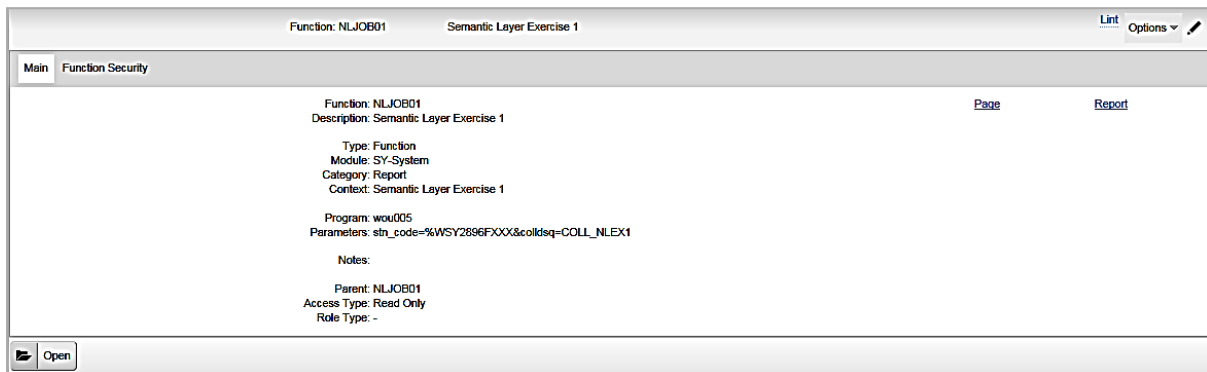
Enter the record selection criteria. Yellow fields are mandatory. Note that the Company Number is not mandatory and will default to the Companies you have access to. This is a key difference to Standard OA Reports, which mostly run for a single COINS Company. Click



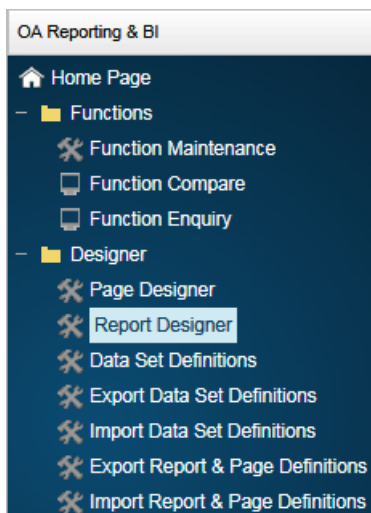
The report will be available on the Report Status Workbench



For example:



In the OA Reporting and BI Module, select the Designer folder and select the Report Designer option

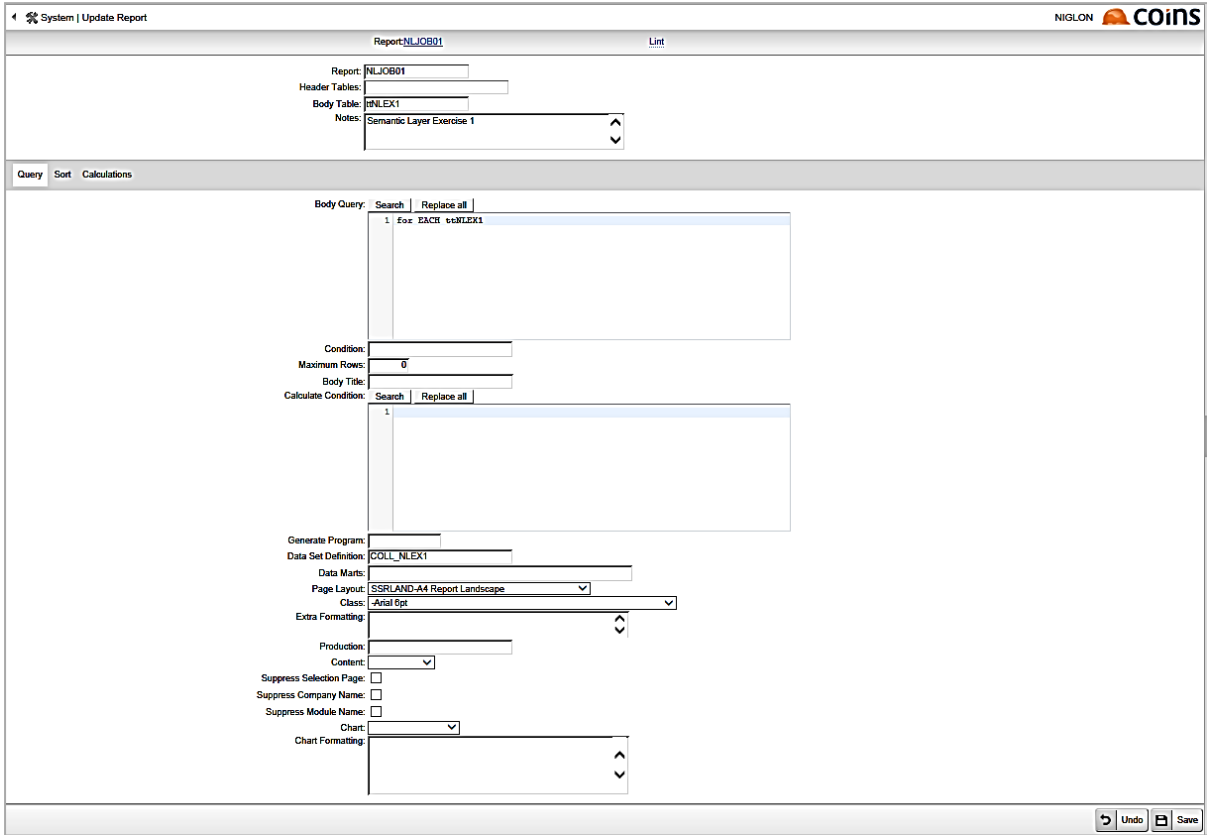


Add a new Report with the same name as the Function created above.

The body table will be ttNLEX1 (i.e. the name of the dataset prefixed with tt).

The body query will be: FOR EACH ttNLEX1

The Data Set Definition will be COLL\_NLEX1 (i.e. the Query Name of the Dataset)



Click  Save

Add a Body Form and add the required fields to it.



Report: NLJOB01 Lint

Header Tables: Function  
 Body Table: #NLEX1  
 Notes: Semantic Layer Exercise 1

Query Sort Calculations Forms Fields Used in

Field	Label	Width	Height	Function	View As
#NLEX1.KCO	Company Number	15mm			
#NLEX1.JOB_NUM	Cost Code Contract	10mm			
#NLEX1.JOB_NAME	Contract Name	30mm			
#NLEX1.JOB_SHIPADDR__1	Contract Address	20mm			
#NLEX1.JOB_SHIPADDR__2	Contract Address	20mm			
#NLEX1.JOB_SHIPADDR__3	Contract Address	20mm			
#NLEX1.JOB_SHIPADDR__4	Contract Address	20mm			
#NLEX1.JOB_PCDE	Contract Postcode	10mm			
#NLEX1.JOB_MANDESC	Contract Manager Name	10mm			
#NLEX1.JOB_COSTSTP	Contract Costs This Period	20mm			

Q+ Advanced Search: Field

Selectors Form: Body View: Standard

Open

When the report is run, the Dataset selection screen is presented. Mandatory fields are in Yellow. Note that the Company Number is not mandatory and will default to the Companies you have access to. This is a key difference to Standard OA Reports, which mostly run for a single COINS Company.

System | Semantic Layer Exercise 1 NIGLON COINS

Main Output Options

Financial Date:

Currency Number:

Company Number: From  To  Matches

<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>

Undo Next

When run, the output will be available from the Report Status Workbench



**System - Semantic Layer Exercise 1**  
**COINS**



Company Number	Cost Code Contract	Contract Name	Contract Address	Contract Address	Contract Address	Contract Address	Contract Postcode	Contract Manager Name	Contract Costs This Period
1	10000	NORTHERN TRUST-P.M. ON HEATERS	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	4,083.00
1	11000	TRAVELCLICK-SHUTDOWN	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	4,029.79
1	11828	ARGONNE 101 - HEAT EXCHANGER	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	0.00
1	11919	111 E. CHESTNUT-BOILER CLEAN	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	4,219.00
1	11994	VERIZON/ZM C#214-COND. UNIT	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	0.00
1	11995	HSA 180 WACKER-VAV & DUCTWORK	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	0.00
1	12032	1212 LASALLE-COMPRESSOR REPL	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	0.00
1	12039	NIELSEN/MASSEY-HEAT EXCHANGERS	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	0.00
1	12046	LJ SHERIDAN-VAV BOX	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	0.00
1	12062	DREYER MEDICAL-HEAT EXCHANGER	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	0.00
1	12078	WASH MUTUAL-COOLER RENTAL	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	0.00
1	12080	JOHN AMANN-HEAT EXCHANGER	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	0.00
1	12082	FACTV - RTU #8 REPAIRS	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	0.00
1	12083	AIM REALTY-INSTALL BOILER	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	0.00
1	12086	FACTV - TURN DIFFUSERS	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	0.00
1	12087	WESTSIDE HOL-HEAT PUMP REPL	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	0.00
1	12088	VVC069 CORP WOODS-RTU REPAIRS	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	0.00
1	12090	TELLABS/ BLDG-RTU REPAIRS	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	0.00
1	12091	VVC740 CORP. WOODS-RAIN HOOD	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	0.00
1	12093	CONFERENCE PLUS-LIEBERT SPLITS	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	0.00
1	12094	FIRST HEALTH-CHILLER REPAIRS	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	0.00
1	12096	WASHINGTON MUTL-SPOT COOLER RN	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	-1,000.00
1	12099	SUN CHEMICAL-REPLACE BARREL	1 Place	Norwich	Norfolk		NR1 B01	Alex Smyrt	0.00

Printed using COINS V10.20.131001SP93053-28/02/14 by Nigel Longley at 15:16:04 on 21/03/14 (NLJ081)

Page 1

## 6.2.4 Exercise – Adding Calculated Fields to a Collection

In the last exercise, we created new fields based on a Token and modified parameters. In this exercise, we are going to create a number of fields using calculations to control their functionality. If performing this exercise, familiarity with either OA Designer, or with Excel formulas is very desirable because this exercise requires the entry of formulas.

Please note that the calculations used are Progress Code and NOT the calculation syntax used elsewhere in the OA BI Toolset. Refer to Appendix of common formulas

### 6.2.4.1 Add the fields

Our dataset currently reports on Costs for each contract, but we now want to break the costs down to each costs category for the current period. The costs Token did not offer a parameter to do this so we need to make use of the Calculated Fields Tokens.

Navigate to the OA Reporting and BI module and select the Semantic Layer folder.

Select the Collections menu option.



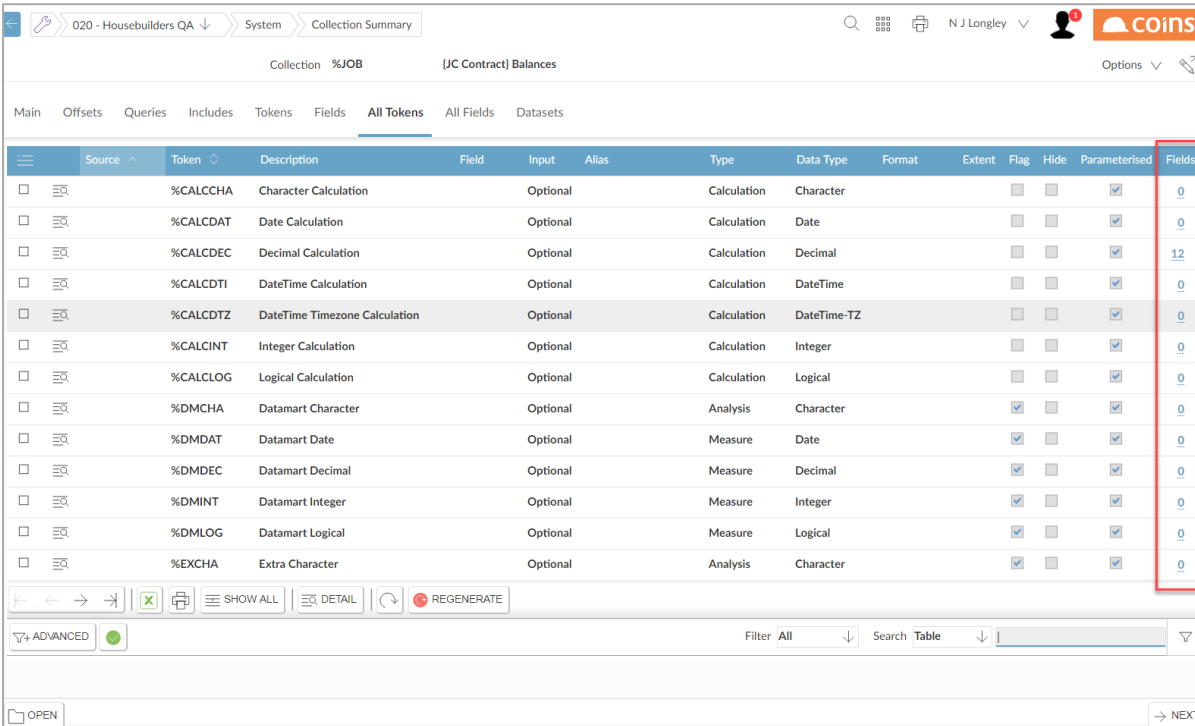
Collection	Description	Tags	Alias
%AINAVM	PL Invoices By Supplier No PO	Financials	Invoice
%AINAVMPO	PL Invoices By Supplier with PO	Financials	Invoice
%AINJOB	PL Invoices By Contract No PO	Financials	Invoice
%AINJOBPO	PL Invoices By Contract with PO	Financials	Invoice
%AVM	Supplier Summary	Financials	Supplier
%AVMGLPER	Supplier Summary By GL Period	Financials	Supplier
%BPF	Business Planning and Forecasting	Commercials, Financials	Contract
%CIM	Company Exposure	Financials, Procurement	Company
%CSTJOB	Sales Certs By (JC Contract)	Financials	CSCertsByContract
%CSTRCM	Sales Certs By Customer	Financials	CSCertsByCustomer
%GLA	General Ledger Accounts	Financials	GLAccount
%GLT	GL Transactions	Financials	GLTransaction
%HCM	Calls	House Building	Call
%HIS	Issues	House Building	Issue
%HTA	Tasks	House Building	Task
%HVI	[Hs Prospects]	House Building	Prospect
%HVIREQDEV	[Hs Prospect] Required (Vp Site)	House Building	Prospect

Select the hyperlink on the %JOB Collection and select the All Tokens Tab.

Set the Search filter to Token and search for \*calc\*

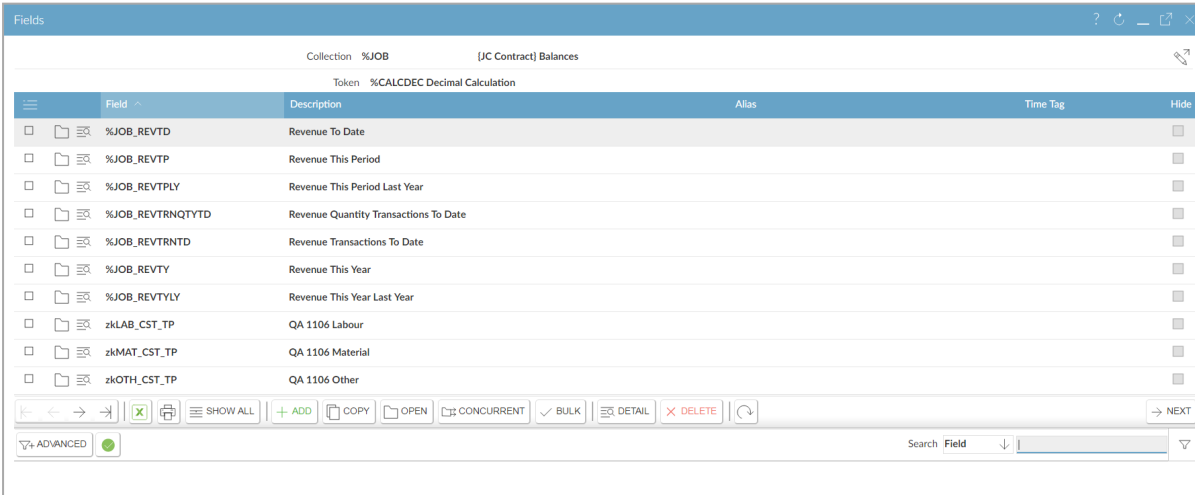
Source	Token	Description	Field	Input	Alias	Type	Data Type	Format	Extent	Flag	Hide	Parameterised	Fields
	%CALCCHA	Character Calculation		Optional		Calculation	Character						0
	%CALCDAT	Date Calculation		Optional		Calculation	Date						0
	%CALCDEC	Decimal Calculation		Optional		Calculation	Decimal						12
	%CALCDTI	DateTime Calculation		Optional		Calculation	DateTime						0
	%CALCDTZ	DateTime Timezone Calculation		Optional		Calculation	DateTime-TZ						0
	%CALCINT	Integer Calculation		Optional		Calculation	Integer						0
	%CALCLOG	Logical Calculation		Optional		Calculation	Logical						0
	%DMCHA	Datamart Character		Optional		Analysis	Character						0
	%DMDAT	Datamart Date		Optional		Measure	Date						0
	%DMDEC	Datamart Decimal		Optional		Measure	Decimal						0
	%DMINT	Datamart Integer		Optional		Measure	Integer						0
	%DMLOG	Datamart Logical		Optional		Measure	Logical						0
	%EXCHA	Extra Character		Optional		Analysis	Character						0

There are calculated field Tokens for each of the data types (Integer, Decimal, Character etc.). The Fields column on the far right indicates the number of fields that have been created against a token and provides a hyperlink to the field definitions.



Source	Token	Description	Field	Input	Alias	Type	Data Type	Format	Extent	Flag	Hide	Parameterised	Fields
	%CALCCHA	Character Calculation		Optional		Calculation	Character					<input checked="" type="checkbox"/>	<a href="#">0</a>
	%CALCDAT	Date Calculation		Optional		Calculation	Date					<input checked="" type="checkbox"/>	<a href="#">0</a>
	%CALCDEC	Decimal Calculation		Optional		Calculation	Decimal					<input checked="" type="checkbox"/>	<a href="#">12</a>
	%CALCDTI	DateTime Calculation		Optional		Calculation	DateTime					<input checked="" type="checkbox"/>	<a href="#">0</a>
	%CALCDTZ	DateTime Timezone Calculation		Optional		Calculation	DateTime-TZ					<input checked="" type="checkbox"/>	<a href="#">0</a>
	%CALCINT	Integer Calculation		Optional		Calculation	Integer					<input checked="" type="checkbox"/>	<a href="#">0</a>
	%CALCLOG	Logical Calculation		Optional		Calculation	Logical					<input checked="" type="checkbox"/>	<a href="#">0</a>
	%DMCHA	Datamart Character		Optional		Analysis	Character			<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	<a href="#">0</a>
	%DMDAT	Datamart Date		Optional		Measure	Date			<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	<a href="#">0</a>
	%DMDEC	Datamart Decimal		Optional		Measure	Decimal			<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	<a href="#">0</a>
	%DMINT	Datamart Integer		Optional		Measure	Integer			<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	<a href="#">0</a>
	%DMLOG	Datamart Logical		Optional		Measure	Logical			<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	<a href="#">0</a>
	%EXCHA	Extra Character		Optional		Analysis	Character			<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	<a href="#">0</a>

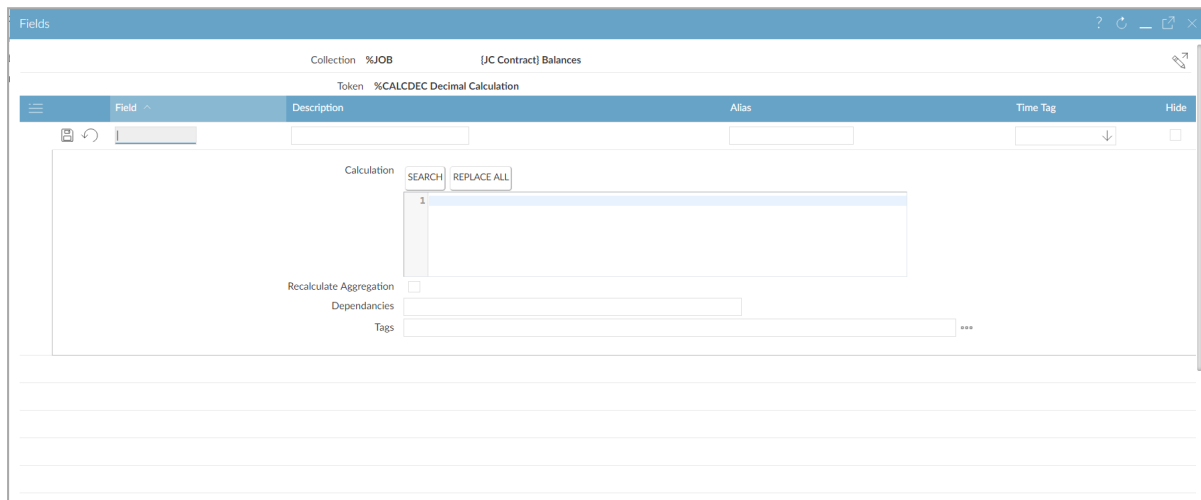
We are going to create fields containing cost figures so select the hyperlink for the fields against %CALCDEC.



Field	Description	Alias	Time Tag	Hide
<a href="#">%JOB_REVTD</a>	Revenue To Date			<input type="checkbox"/>
<a href="#">%JOB_REVTP</a>	Revenue This Period			<input type="checkbox"/>
<a href="#">%JOB_REVTPPLY</a>	Revenue This Period Last Year			<input type="checkbox"/>
<a href="#">%JOB_REVTNRQTYTD</a>	Revenue Quantity Transactions To Date			<input type="checkbox"/>
<a href="#">%JOB_REVTNRNTD</a>	Revenue Transactions To Date			<input type="checkbox"/>
<a href="#">%JOB_REVTY</a>	Revenue This Year			<input type="checkbox"/>
<a href="#">%JOB_REVTYLY</a>	Revenue This Year Last Year			<input type="checkbox"/>
<a href="#">zkLAB_CST_TP</a>	QA 1106 Labour			<input type="checkbox"/>
<a href="#">zkMAT_CST_TP</a>	QA 1106 Material			<input type="checkbox"/>
<a href="#">zkOTH_CST_TP</a>	QA 1106 Other			<input type="checkbox"/>

The %JOB Collection contains links to Contract data and Cost Code data and we can reference this in our calculations.

Click



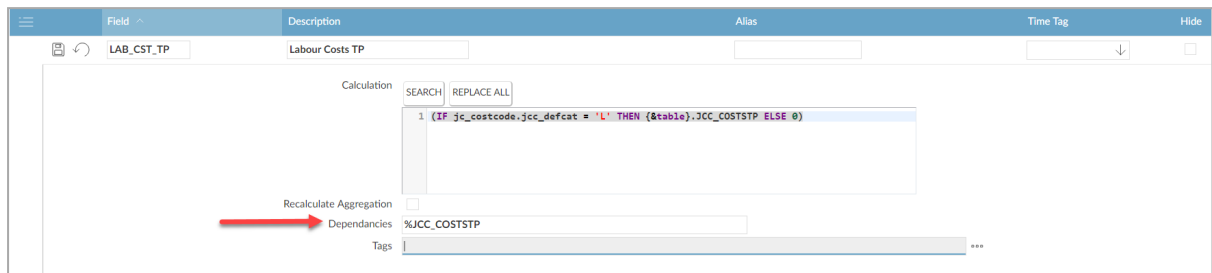
Our first field is going to breakdown Labour Costs, so give the field a name (for example xxLAB\_CST\_TP where xx = your initials) and a description that will be meaningful to your users.

Our calculation needs to look at the Category code against each cost code and add up the costs to date if it matches 'L', (which is the category code for Labour in the COINS training database – your COINS system may use a different code). There is already a field in the Collection for accumulating Cost Code Costs for this period (%JCC\_COSTSTP) so we can use this as the basis to accumulate our broken down costs

Our calculation (in Progress syntax) is as follows.

```
(IF jc_costcode.jcc_defcat = 'L' THEN {&table}.JCC_COSTSTP ELSE 0)
```

The calculation requires %JCC\_COSTSTP in order to function, so we need to ensure that it is automatically added to any datasets. We can do this by adding an entry for %JCC\_COSTSTP to the Dependencies field.



Click

Copy the field and change its name to xxMAT\_CST\_TP with a description appropriate for Material Costs.

Change the Calculation to look for a Category code of M (Materials)



Click

Repeat from step 11 for category codes A (Prelims), O (Other), P (Plant) and S(Subcon)

Field	Description	Alias	Time Tag	Hide
LAB_CST_TP	Labour Costs TP			<input type="checkbox"/>
MAT_CST_TP	Material Costs TP			<input type="checkbox"/>
OTH_CST_TP	Other Costs TP			<input type="checkbox"/>
PLA_CST_TP	Plant Costs TP			<input type="checkbox"/>
PRE_CST_TP	Prelim Costs TP			<input type="checkbox"/>
SUB_CST_TP	Subcon Costs TP			<input type="checkbox"/>

#### 6.2.4.2 Add the fields to the Dataset

Once the fields are created in the collection, they are immediately available to selection in new or existing Datasets. In this exercise we will add them to our existing dataset xxEX1.

In Collection %JOB, navigate to the Datasets Tab



Dataset	Description	ODBC Name	Query Name	Alias	Program	Compiled
NLEX1	Contract Details				06/12/2016 08:48:06.000	06/12/2016 08:48:06.000

Open the Dataset xxEX1 and add your two new fields to the dataset

Collection: %JOB (JC Contract) Balances  
 Dataset: NLEX1 Contract Details  
 ODBC Name:   
 Query Name: COLL\_NLEX1  
 Alias:   
 Program: 06/12/2016 08:48:06.000  
 Compiled: 06/12/2016 08:48:06.000

Fields Available:

- Contract Revenue Quantity (CS) Transactions To Date (%JOB\_REVTRNQTY)
- Contract CS Gross Take On Value (%JOB\_CSTKONGRS)
- Contract CS Net Take On Value (%JOB\_CSTKONNET)
- Contract CS VAT Take On Value (%JOB\_CSTKONVAT)
- Contract Revenue (CS) This Period (%JOB\_REVCSTP)
- Contract Revenue (CS) This Period Last Year (%JOB\_REVCSTPLY)
- Contract Revenue (CS) This Year (%JOB\_REVCSTY)
- Contract Revenue (CS) This Year Last Year (%JOB\_REVCSTLY)
- Contract Revenue (CS) To Date (%JOB\_REVCSTD)
- Contract Revenue (CS) Transactions To Date (%JOB\_REVTRNCSTD)
- QA 1106 Material (zkMAT\_CST\_TP)
- QA 1106 Other (zkOTH\_CST\_TP)
- QA 1106 Plant (zkPLA\_CST\_TP)
- QA 1106 Subcon (zkSUB\_CST\_TP)
- Subcon Costs TP (SUB\_CST\_TP)

Selected:

- Currency Number (%INP\_CUR\_NUM)
- Financial Date (%INP\_GLP\_FDATE)
- Company Number (%KCO)
- Contract (%JOB\_NUM)
- Contract Name (%JOB\_NAME)
- Contract Address (%JOB\_SHIPADDR)
- Contract Postcode (%JOB\_PCODE)
- Contract Manager Name (%JOB\_MANDESC)
- Contract Costs This Period (%JOB\_COSTSTP)
- Labour Costs TP (LAB\_CST\_TP)
- Material Costs TP (MAT\_CST\_TP)
- Other Costs TP (OTH\_CST\_TP)
- Plant Costs TP (PLA\_CST\_TP)
- Prelim Costs TP (PRE\_CST\_TP)
- QA 1106 Labour (zkLAB\_CST\_TP)

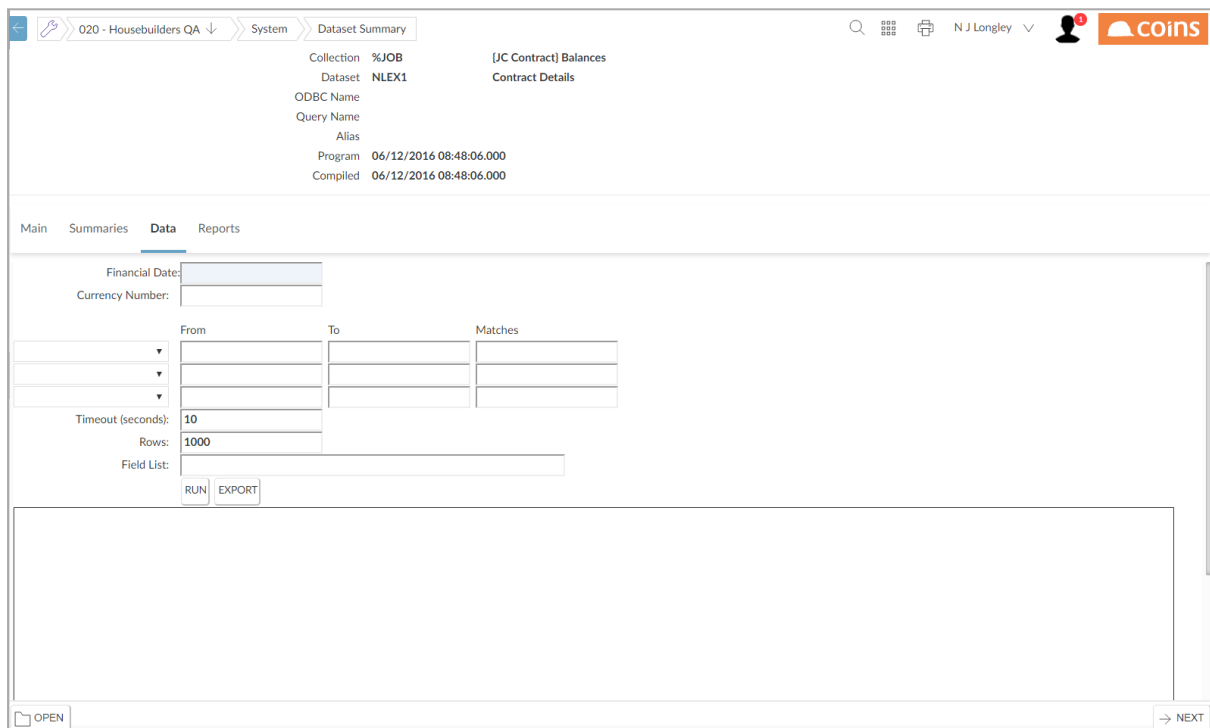
Click

Check the Program and Compiled date and Time have updated

At this point, your dataset will have the new fields and any new reports based on this dataset will have them available for selection. Any existing reports based on this dataset will have the columns available for adding next time you edit the design but will not automatically have the columns added.

### 6.2.4.3 Test the Dataset

Select the Data tab



The coloured field indicates where a mandatory value is required – in this case it is the Financial Period on which we want to report. Enter a valid Financial Date and click **Run**

Your new fields should now appear on the output with the figures changing as appropriate.



020 - Housebuilders QA System Dataset Summary

Collection %JOB [JC Contract] Balances  
 Dataset NLEX1 Contract Details  
 ODBC Name  
 Query Name COLL\_NLEX1  
 Alias  
 Program 06/12/2016 15:54:55.000  
 Compiled 06/12/2016 15:54:56.000

Main Summaries **Data** Reports

Financial Date: 31/01/16  
 Currency Number:

From To Matches

Timeout (seconds): 10  
 Rows: 1000  
 Field List:

RUN EXPORT

DDR_1	JOB_SHIPADDR_2	JOB_SHIPADDR_3	JOB_SHIPADDR_4	JOB_PC	JOB_MANDESC	JOB_COSTSTP	LAB_CST_TP	MAT_CST_TP	OTH_CST_TP	PLA_CST_TP	PRE_CST_TP	LAB_CST_TP	JCC_COSTSTP
					Lana Bayda	0	0	0	0	0	0	0	0
				197 547	Alex Smyrl	100	0	100	0	0	0	0	100
						0	0	0	0	0	0	0	0
						100	0	0	0	0	0	0	100
						0	0	0	0	0	0	0	0
1 & 2	Anchor Boulevard, Crossways	Dartford, Kent	United Kingdom	DA2 6SN	Steve Jones	0	0	0	0	0	0	0	0

OPEN NEXT

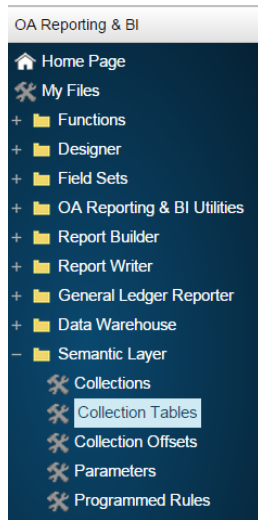
### 6.2.5 Exercise - Adding Fields to a Collection Table

In exercise 3 we added a field to a collection. It should be emphasised that only if the field is relative to another field, the value of which is not known until the collection; for instance a field which relies on the Financial Date (as in the previous examples), should it be created on the collection.

If the field is relative to the table only then it should be created on the table as a field. In this way it will be available to any collection that uses that table in the query. By creating a field on the collection it is only available to that collection.




### 6.2.5.1 Add a new Token



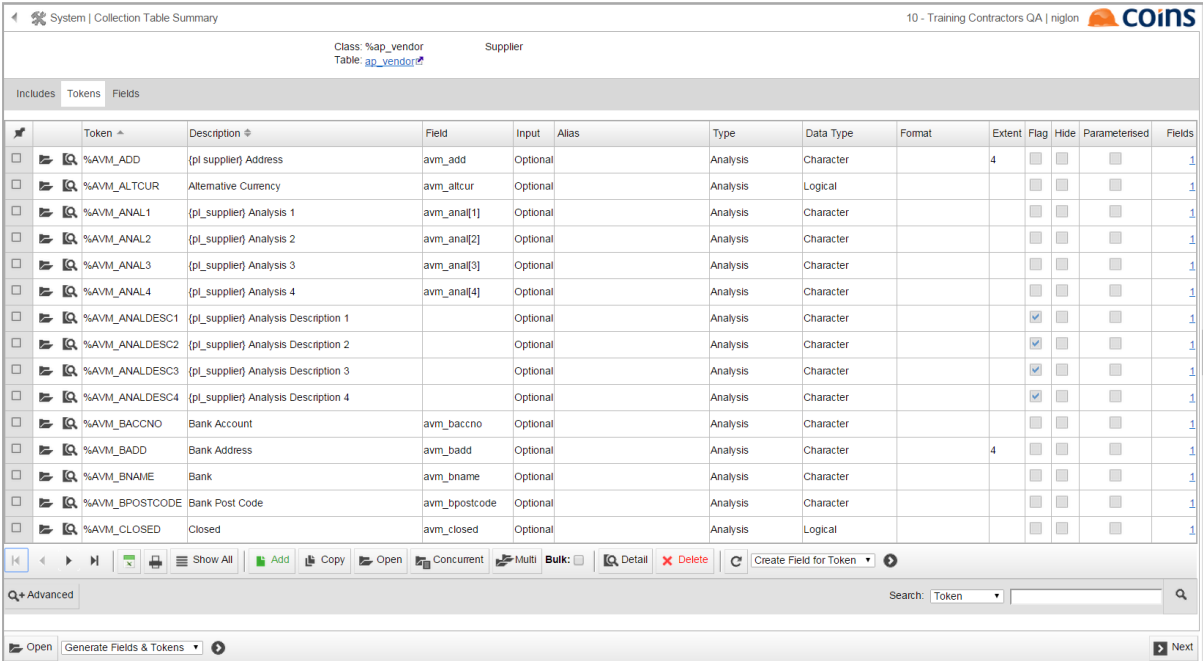
Similar to RO fields, Tokens are Progress code subroutines that collect data. For this exercise we are going to create a token to check the hold code on supplier records and if a supplier is on-hold, we are going to generate a field that will contain the text “ON-HOLD” that we can use in a report.

Navigate to the OA Reporting and BI module and select the Semantic Layer folder.

Select the Collection Tables menu option

System   Collection Tables		10 - Training Contractors QA   nlgton 	
Class	Description	Table	
%s	Generic Tokens		
%sap_vendor	Supplier	ap_vendor	
%ar_cust	Customer Details	ar_cust	
%ar_invdist	Sales Ledger Distribution	ar_invdist	
%ar_invoice	Sales Ledger Invoices	ar_invoice	
%cl_company	External Company	cl_company	
%cl_office	Office of External Company	cl_office	
%cm_component	Item of Equipment	cm_component	
%cm_costcode	EQ Cost Code	cm_costcode	
%co_config	{Kco Company}	co_config	
%gl_acct	General Ledger Accounts	gl_acct	
%gl_period	GL Period	gl_period	
%gl_trans	GL transactions	gl_trans	
%hs_complaint	Call	hs_complaint	
%hs_issue	Issue	hs_issue	
%hs_task	Task	hs_task	
%hs_transREQDEV	Required (Vp Site)	hs_trans	
%hs_visitor	{Hs Prospect}	hs_visitor	
%jc_costcode	Cost Code	jc_costcode	
%jc_costcodeCM	Commercials Cost Codes	jc_costcode	

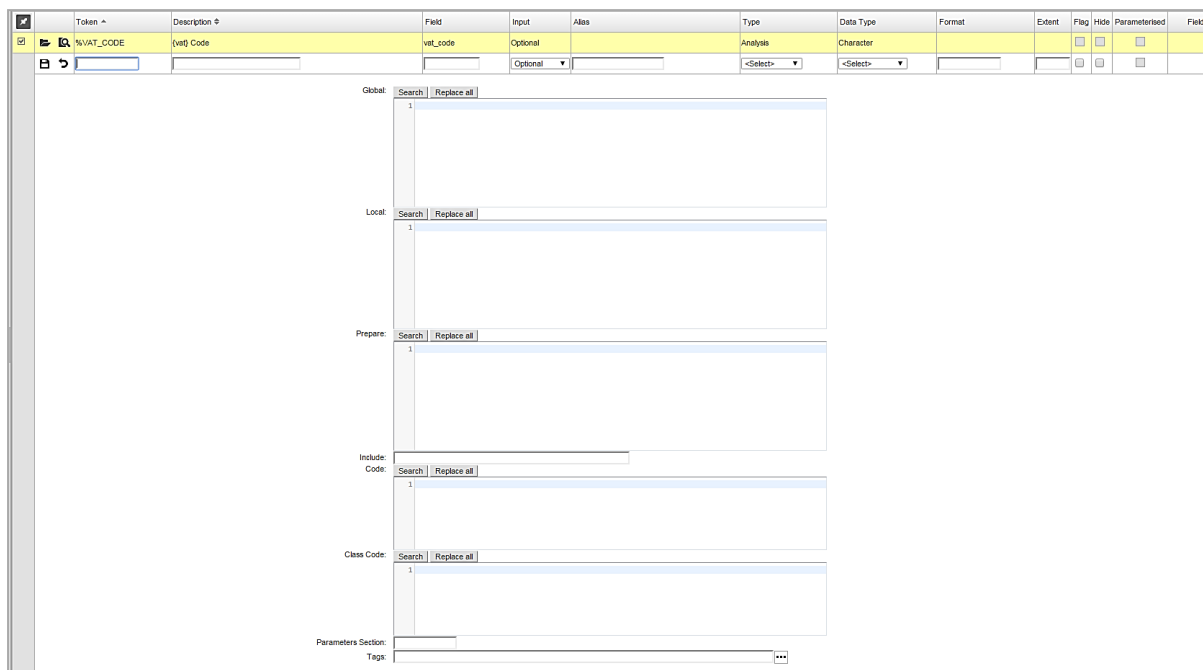
Select the hyperlink on the table %ap\_vendor and select the Tokens tab.



Token	Description	Field	Input	Alias	Type	Data Type	Format	Extent	Flag	Hide	Parameterised	Fields
%AVM_ADD	{pl_supplier} Address	avm_add	Optional		Analysis	Character		4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1
%AVM_ALTCUR	Alternative Currency	avm_altcur	Optional		Analysis	Logical			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1
%AVM_ANAL1	{pl_supplier} Analysis 1	avm_anal[1]	Optional		Analysis	Character			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1
%AVM_ANAL2	{pl_supplier} Analysis 2	avm_anal[2]	Optional		Analysis	Character			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1
%AVM_ANAL3	{pl_supplier} Analysis 3	avm_anal[3]	Optional		Analysis	Character			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1
%AVM_ANAL4	{pl_supplier} Analysis 4	avm_anal[4]	Optional		Analysis	Character			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1
%AVM_ANALDESC1	{pl_supplier} Analysis Description 1		Optional		Analysis	Character			<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1
%AVM_ANALDESC2	{pl_supplier} Analysis Description 2		Optional		Analysis	Character			<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1
%AVM_ANALDESC3	{pl_supplier} Analysis Description 3		Optional		Analysis	Character			<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1
%AVM_ANALDESC4	{pl_supplier} Analysis Description 4		Optional		Analysis	Character			<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1
%AVM_BACCNO	Bank Account	avm_baccno	Optional		Analysis	Character			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1
%AVM_BADD	Bank Address	avm_badd	Optional		Analysis	Character		4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1
%AVM_BNAME	Bank	avm_bname	Optional		Analysis	Character			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1
%AVM_BPOSTCODE	Bank Post Code	avm_bpostcode	Optional		Analysis	Character			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1
%AVM_CLOSED	Closed	avm_closed	Optional		Analysis	Logical			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1

If we look through the available Tokens, we can find %AVM\_HOLD which has a single field that returns the value of the Hold field on the supplier record. This will be a Logical Y or N value. We want to create a Token that will take this Y or N value and create a field that will contain either blank or “ON-HOLD”

Click Add



Give the Token a name of HOLD\_TEXT, do not prefix with % as this is not a standard Token and we do not want it overwritten in future updates from COINS. Token names should be named similarly to the fields they represent in the database.

The token name and description should make it obvious to a designer what the token will do.

Add description of On Hold Report Text.

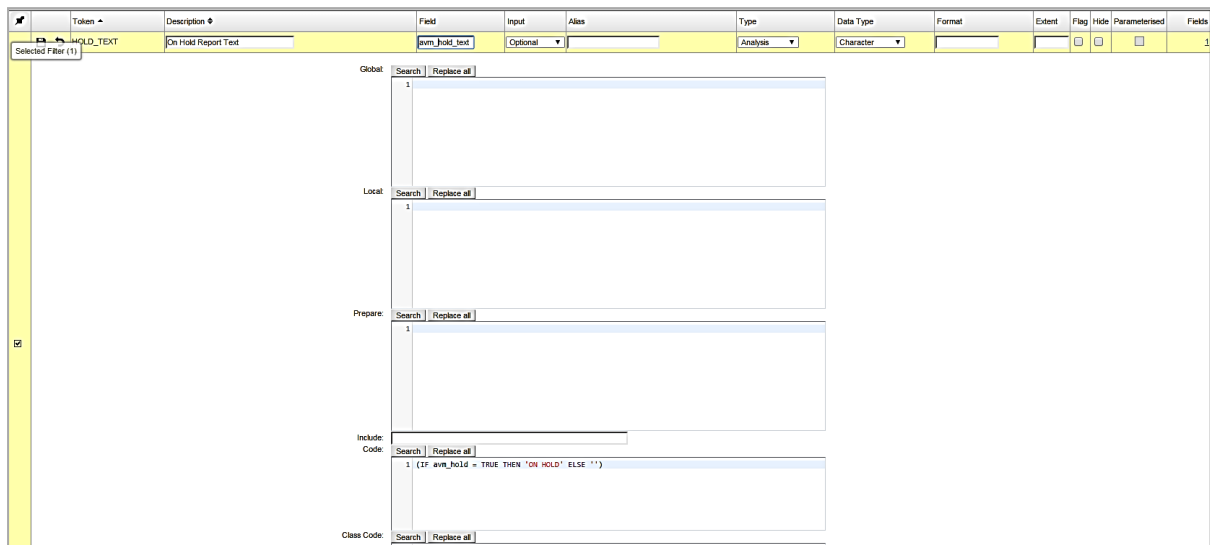
Add a field name of hold\_text

Set Input to Optional

Set Type to Analysis, and Data Type to Character.

We now need to add some progress code that tells the Token how to generate field values. In the Code section add the code:

```
(IF avm_hold = TRUE THEN 'ON HOLD' ELSE '')
```



Click Save.

Select the new field and on the Action Dropdown select Generate Fields & Tokens and click Apply.

If there are any errors in the Progress code, these will be indicated in a dialog box. These errors will need to be corrected before the Token/field can be created. If there are no errors, the Token will be created and the Fields column will change from 0 to 1.

Navigate back to Collections and open the %AVM collection.

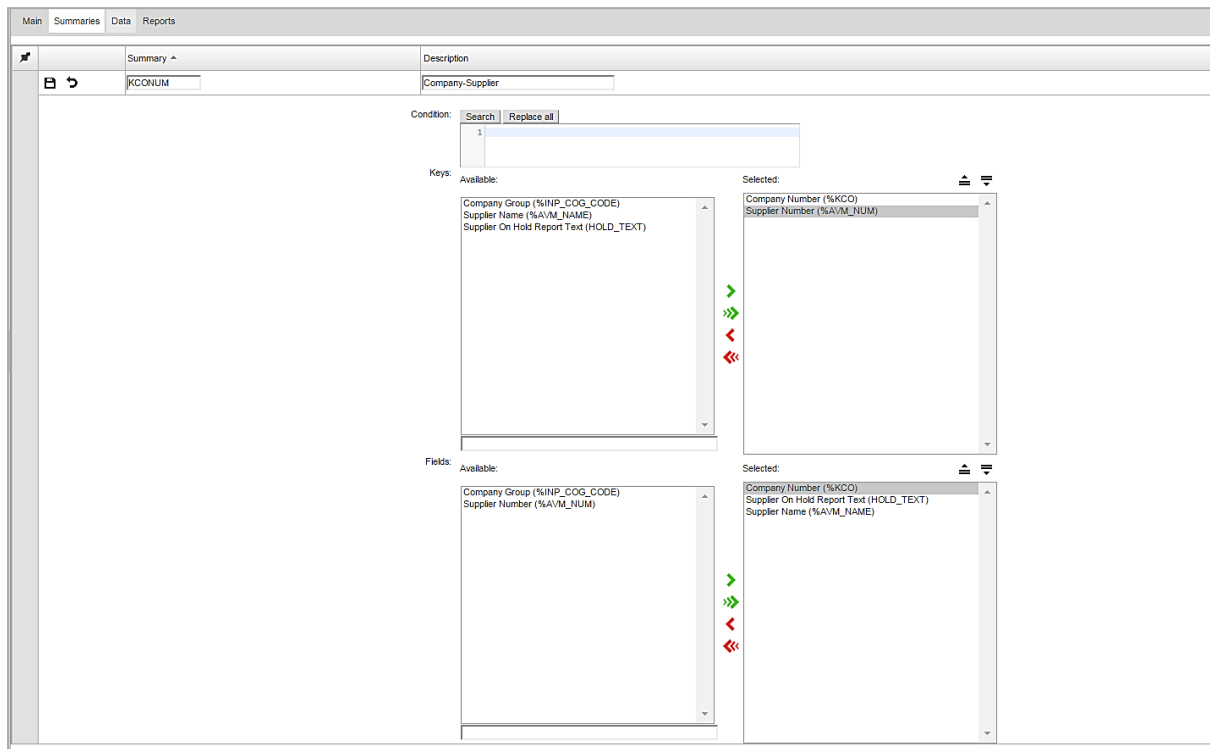
Create a new dataset called Exercise5xx where xx = your initials. E.g. Exercise5NL

Open the Main Tab and add the fields for Company number (KCO), Supplier Number (%AVM\_NUM), Supplier Name (%AVM\_NAME) and Supplier On Hold Report Text (HOLD\_TEXT)

Click Save

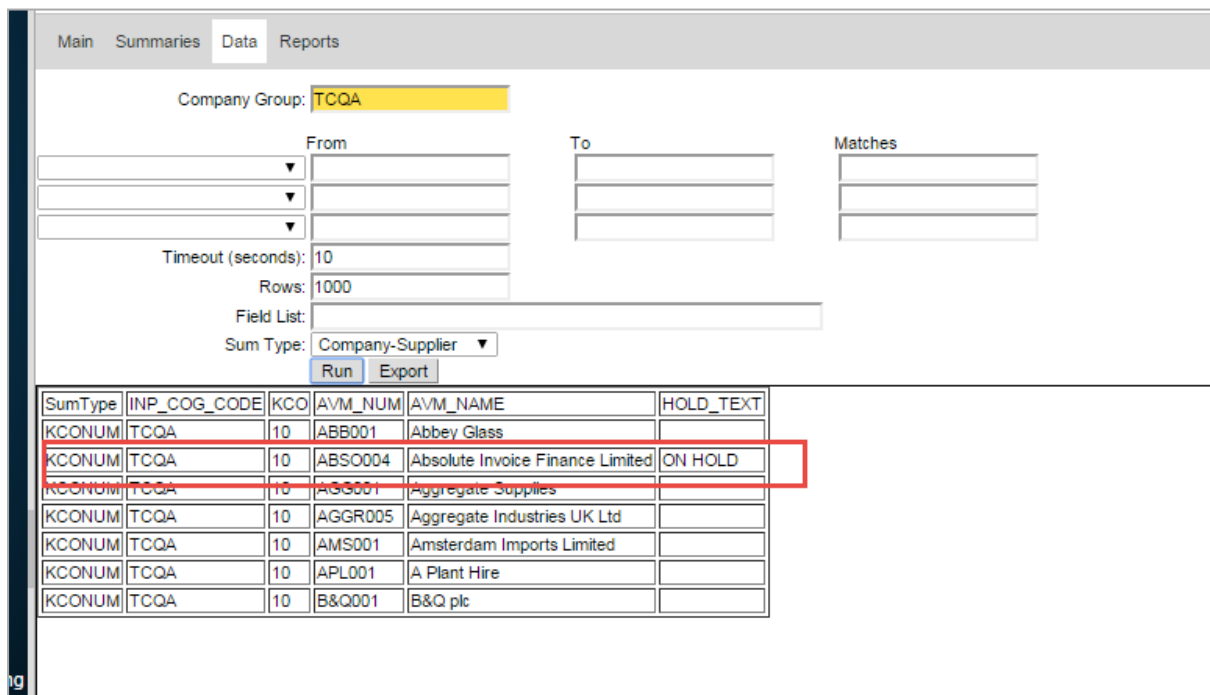
Create a Summary called KCONUM, with Keys of Company Number (%KCO) and Supplier Number (%AVM\_NUM)

Add the fields for Company Number(%KCO), Supplier Name (%AVM\_NAME) and Supplier On Hold Report Text (HOLD\_TEXT)



On the Data Tab, enter the Company Group TCQA (this is the company group in the COINS training environment – your system will have different Company Group names)

Set the Sum Type to Company-Supplier and click Run

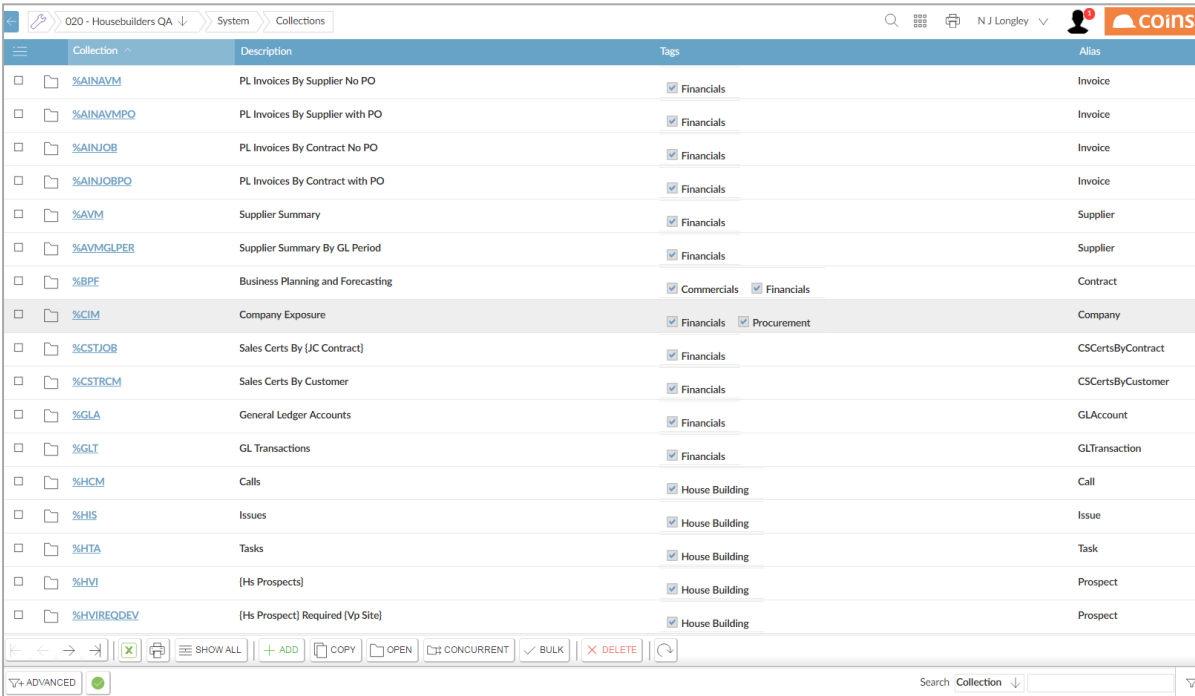


## 6.2.6 Exercise – Adding fields to a Collection

In this exercise we are going to add create some additional fields within the %JOB collection and add them to the dataset we created in Exercise 1

### 6.2.6.1 Add the fields

- Navigate to **OA Reporting and BI > Semantic Layer > Collections**



Collection	Description	Tags	Alias
<a href="#">%AINAVM</a>	PL Invoices By Supplier No PO	<input checked="" type="checkbox"/> Financials	Invoice
<a href="#">%AINAVMPO</a>	PL Invoices By Supplier with PO	<input checked="" type="checkbox"/> Financials	Invoice
<a href="#">%AINJOB</a>	PL Invoices By Contract No PO	<input checked="" type="checkbox"/> Financials	Invoice
<a href="#">%AINJOBPO</a>	PL Invoices By Contract with PO	<input checked="" type="checkbox"/> Financials	Invoice
<a href="#">%AVM</a>	Supplier Summary	<input checked="" type="checkbox"/> Financials	Supplier
<a href="#">%AVMGLPER</a>	Supplier Summary By GL Period	<input checked="" type="checkbox"/> Financials	Supplier
<a href="#">%BPFE</a>	Business Planning and Forecasting	<input checked="" type="checkbox"/> Commercials <input checked="" type="checkbox"/> Financials	Contract
<a href="#">%CIM</a>	Company Exposure	<input checked="" type="checkbox"/> Financials <input checked="" type="checkbox"/> Procurement	Company
<a href="#">%CSTJOB</a>	Sales Certs By (JC Contract)	<input checked="" type="checkbox"/> Financials	CSertsByContract
<a href="#">%CSTRCM</a>	Sales Certs By Customer	<input checked="" type="checkbox"/> Financials	CSertsByCustomer
<a href="#">%GLA</a>	General Ledger Accounts	<input checked="" type="checkbox"/> Financials	GLAccount
<a href="#">%GLT</a>	GL Transactions	<input checked="" type="checkbox"/> Financials	GLTransaction
<a href="#">%HCM</a>	Calls	<input checked="" type="checkbox"/> House Building	Call
<a href="#">%HIS</a>	Issues	<input checked="" type="checkbox"/> House Building	Issue
<a href="#">%HTA</a>	Tasks	<input checked="" type="checkbox"/> House Building	Task
<a href="#">%HVI</a>	[Hs Prospects]	<input checked="" type="checkbox"/> House Building	Prospect
<a href="#">%HVIREQDEV</a>	[Hs Prospect] Required (Vp Site)	<input checked="" type="checkbox"/> House Building	Prospect

- Select the Hyperlink on the %JOB Collection and select the All Fields Tab



001 - CONTRACTOR MASTER System Collection Summary [JC Contract] Balances

Main Offsets Queries Includes Tokens Fields All Tokens All Fields Datasets

Level	Type	Data Type	Source	Field Description	Time Tag	Tags	Field	Token	Hide	Alias
0	1-Key	Integer	Collection	Currency Number			%INP_CUR_NUM	%INP_CUR_NUM		
0	1-Key	Date	Collection	Financial Date			%INP_GLP_FDATE	%INP_GLP_FDATE		
1	2-Analysis	Character	Company	Name			%COC_NAME	%COC_NAME		
1	2-Analysis	Integer	Company	Number			%KCO	%KCO		
2	1-Key	Character	Contract	Location Description			%JCL_DESC	%JCL_DESC		
2	1-Key	Character	Contract	Location			%JCL_LOC	%JCL_LOC		
2	1-Key	Character	Contract	Group Description			%JGR_DESC	%JGR_DESC		
2	1-Key	Character	Contract	Group			%JGR_GROUP	%JGR_GROUP		
2	1-Key	Character	Contract	Manager			%JOB_MAN	%JOB_MAN		
2	1-Key	Character	Contract	Manager Name			%JOB_MANDESC	%JOB_MANDESC		
2	1-Key	Character	Contract	Name			%JOB_NAME	%JOB_NAME		
2	1-Key	Character	Contract	Contract			%JOB_NUM	%JOB_NUM		
2	1-Key	Character	Contract	Type Description			%JTY_DESC	%JTY_DESC		
2	1-Key	Character	Contract	Type			%JTY_TYPE	%JTY_TYPE		
2	1-Key	Character	Contract	Customer Name			%RCM_NAME	%RCM_NAME		
2	1-Key	Character	Contract	Customer			%RCM_NUM	%RCM_NUM		
2	2-Analysis	Character	Contract	Analysis 1 (Status of Contract)			%JOB_ANAL1	%JOB_ANAL1		
2	2-Analysis	Character	Contract	Analysis 2 ()			%JOB_ANAL2	%JOB_ANAL2		
2	2-Analysis	Character	Contract	Analysis 3 ()			%JOB_ANAL3	%JOB_ANAL3		

ADVANCED Filter: All Search: Field

In our Dataset, we used the field Contract Costs This Period (%JOB\_COSTSTP). However, we have been asked to also report on Costs last period, and the period before that. These were not available within the collection, so we need to add this functionality into the collection before we can add it to our dataset.

Since we already have a field looking at period costs, we can base our new fields on the existing one

- Set the Search Filter to Field and %JOB\_COSTSTP



001 - CONTRACTOR MASTER System Collection Summary [JC Contract] Balances

Main Offsets Queries Includes Tokens Fields All Tokens All Fields Datasets

Level	Type	Data Type	Source	Field Description	Time Tag	Tags	Field	Token	Hide	Alias
4	4-Measure	Decimal	Contract	Costs This Period			%JOB_COSTSTP	%JOB_COSTS	<input type="checkbox"/>	
4	4-Measure	Decimal	Contract	Costs This Period Last Year			%JOB_COSTSTPLY	%JOB_COSTS	<input type="checkbox"/>	

Filter All Search Field %job\_coststp

Against the field, you will see a column called Token. Tokens define the way data in a field is generated. Multiple fields may use the same token, but by altering some parameters, each can change the way in which they use it.

- Select the hyper link against the Token (%JOB\_COSTS).

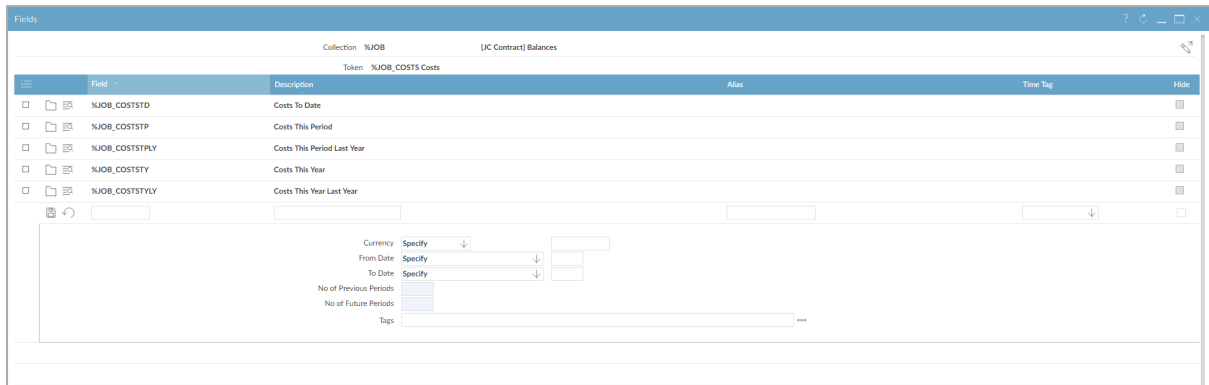
Field	Description	Alias	Time Tag	Hide
%JOB_COSTSTD	Costs To Date			<input type="checkbox"/>
%JOB_COSTSTP	Costs This Period			<input type="checkbox"/>
%JOB_COSTSTPLY	Costs This Period Last Year			<input type="checkbox"/>
%JOB_COSTSTY	Costs This Year			<input type="checkbox"/>
%JOB_COSTSTLY	Costs This Year Last Year			<input type="checkbox"/>

Search Field

This screen lists the fields currently defined by this Token. As we can see, there are no fields defined that look one or two periods back from This Period.

Click





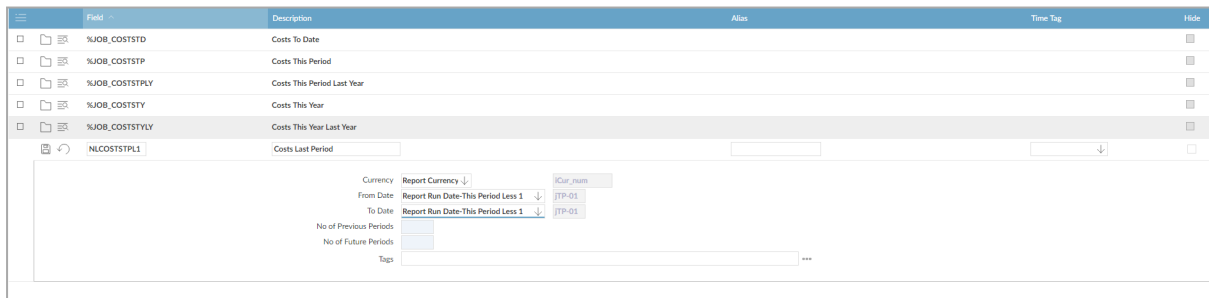
Specify a field name of xxCostsTPL1 (Where xx are your initials) and a description of Costs Last Period.

Set Currency to Report Currency

Set From Date to Report Run Date – This Period Less 1

Set To Date to Report Run Date – This Period Less 1

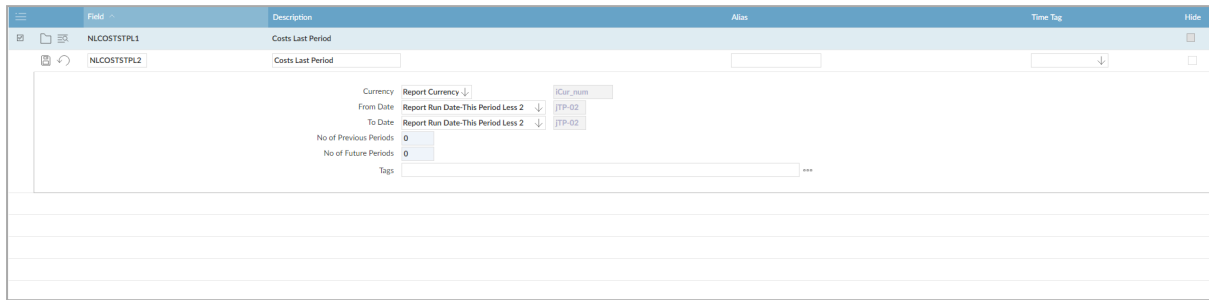
Previous and Future Periods should be set as 0 (Zero)



Click Save

Copy the field you have just created. Change its name to xxCostsTPL2 and its description to Costs This Period Less 2.

Amend the From and To dates to Report Run Date - This Period Less 2

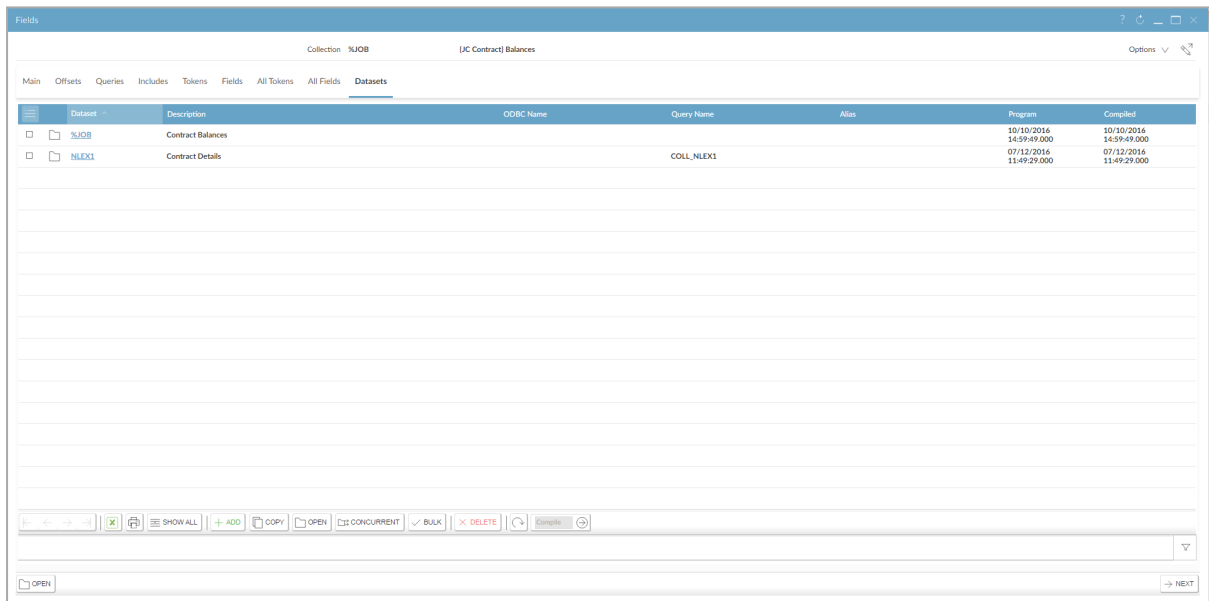


Click Save

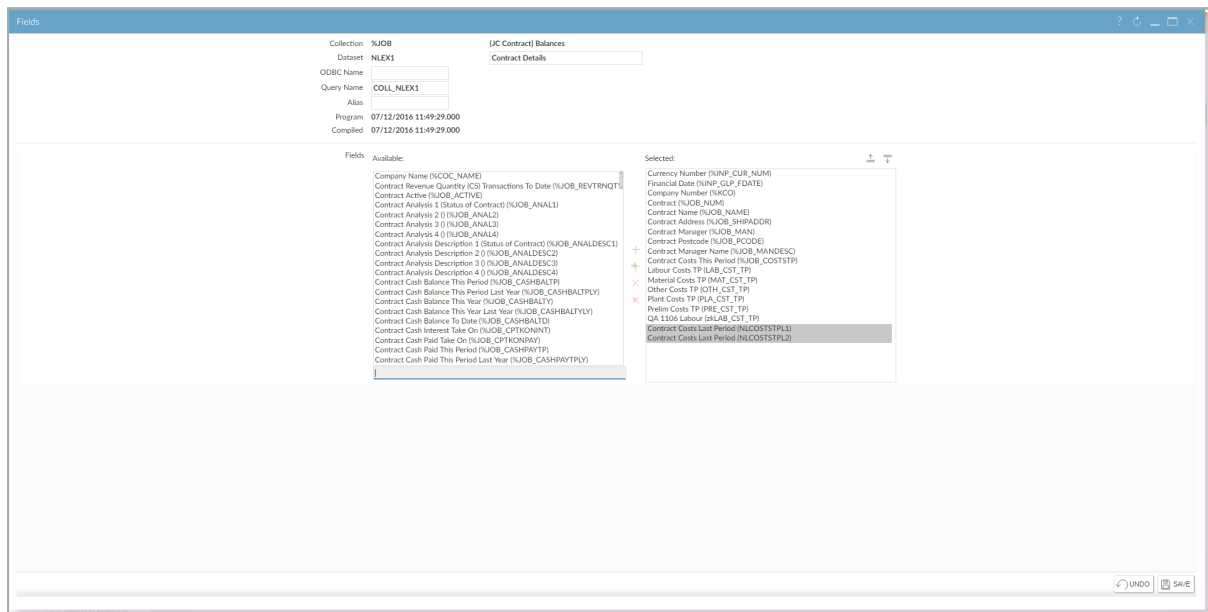
### 6.2.6.2 Add the fields to the Dataset

Once the fields are created in the collection, they are immediately available to selection in new or existing Datasets. In this exercise we will add them to our existing dataset xxEX1.

In Collection %JOB, navigate to the Datasets Tab



Open the Dataset xxEX1 and add your two new fields to the dataset



Click

Check the Program and Compiled date and Time have updated

Collection	%JOB	<b>{JC Contract} Balances</b>
Dataset	NLEX1	<b>Contract Details</b>
ODBC Name		
Query Name	COLL_NLEX1	
Alias		
Program	07/12/2016 14:20:34.000	
Compiled	07/12/2016 14:20:34.000	

If these do not update, your changes will not be effective and you may need to seek assistance to identify why the compile did not take place. As checks are made at all stages of your changes, this should be a rare occurrence.

At this point, your dataset will have the new fields and any new reports based on this dataset will have them available for selection. Any existing reports based on this dataset will have the columns available for adding next time you edit the design but will not automatically have the columns added.

### 6.2.6.3 Test the Dataset

Select the Data tab



020 - Housebuilders QA System Dataset Summary

Collection %JOB [JC Contract] Balances  
Dataset NLEX1 Contract Details  
ODBC Name  
Query Name  
Alias  
Program 06/12/2016 08:48:06.000  
Compiled 06/12/2016 08:48:06.000

Main Summaries **Data** Reports

Financial Date:   
Currency Number:

From To Matches

Timeout (seconds): 10  
Rows: 1000  
Field List:

RUN EXPORT

OPEN NEXT

The coloured field indicates where a mandatory value is required – in this case it is the Financial Period on which we want to report. Enter a valid Financial Date and click **Run**

Your new fields should now appear on the output with the figures changing as appropriate.

Fields

Collection %JOB [JC Contract] Balances  
Dataset NLEX1 Contract Details  
ODBC Name  
Query Name COLL\_NLEX1  
Alias  
Program 07/12/2016 14:20:34.000  
Compiled 07/12/2016 14:20:34.000

Main Summaries **Data** Reports

Financial Date:   
Currency Number:

From To Matches

Timeout (seconds): 10  
Rows: 1000  
Field List:   
Sum Type: All

RUN EXPORT

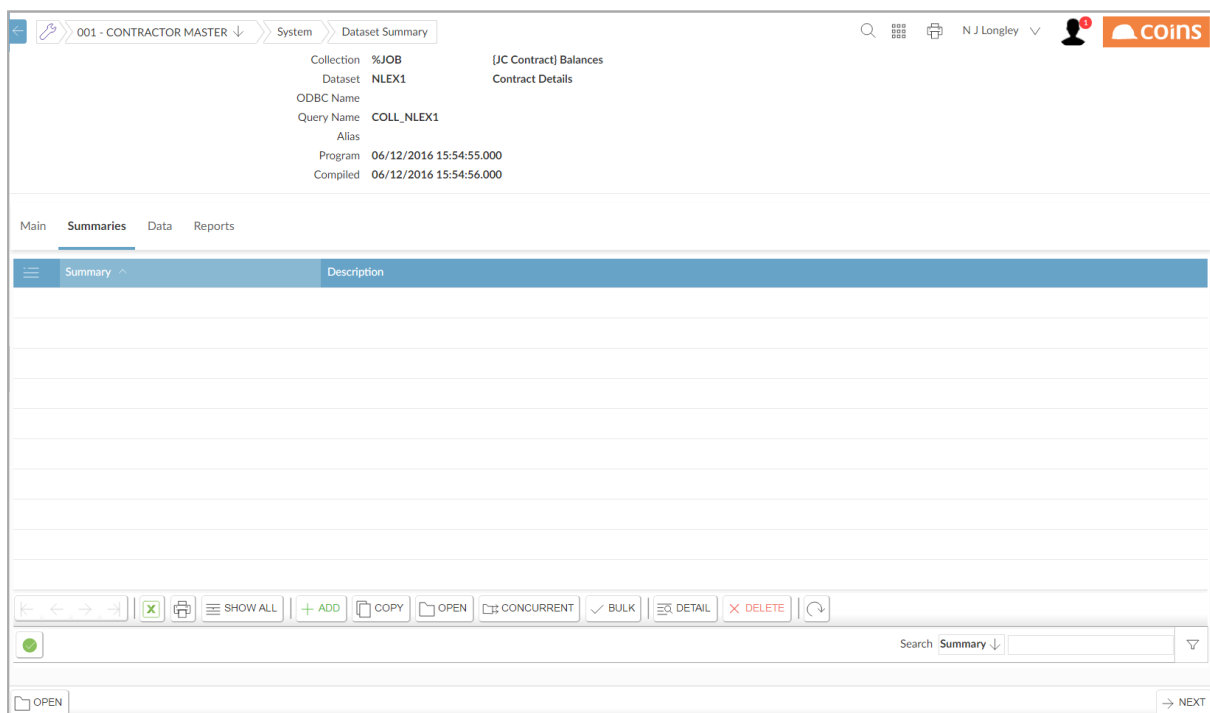
SumType	INP_CUR_NUM	INP_GLP_FDATE	KCO	JOB_NUM	JOB_NAME	JOB_SHIPADDR_1	JOB_SHIPADDR_2	JOB_SHIPADDR_3	JOB_SHIPADDR_4	JOB_MAN	JOB_PCDD	JOB_MANDESC	JOB_COSTSTP	LAB_CST_TP	MAT_CST_TP	OTH_CST_TP	PLA_CST_TP	PRE_CST_TP	SLAB_CST_TP	NLCCOSTSTP1	NLCCOSTSTP2
2		31/01/16	1	000012	ISLEWICK					lanbay		Lana Bayda	0	0	0	0	0	0	0	0	0
2		31/01/16	1	777888	ANTFOM F-1 Clie	lenina st				ALESMY	197 547	Alex Smyth	100	0	100	0	0	0	0	0	0
2		31/01/16	1	AZ001	prozab Contract								0	0	0	0	0	0	0	0	0
2		31/01/16	1	AZ002	AZ002 Contract								100	0	0	0	0	0	0	0	0
2		31/01/16	1	0cstest	0cstest								0	0	0	0	0	0	0	0	0
2		31/01/16	1	c4123	Lang O'Rourke	Bridge Place 1 & 2	Anchor Boulevard, Crossways	Dartford, Kent	United Kingdom	stojon	DA2 6SN	Steve Jones	0	0	0	0	0	0	0	0	0
2		31/01/16	1	c4124	best blane					stojon		Steve Jones	0	0	0	0	0	0	0	0	0
2		31/01/16	1	CANADA	CANADA					decoba		Declan O'Hara	0	0	0	0	0	0	0	0	0
2		31/01/16	1	EB1234	Contract with NIS							4050	0	0	3000	0	0	0	0	0	10
2		31/01/16	1	0de001	0de0 project	28 Green str.	Kullingrad			lanbay	128176	Lana Bayda	0	0	0	0	0	0	0	0	0
2		31/01/16	1	0G01	0gor Test Contract							Lana Bayda	0	0	0	0	0	0	0	0	0

OPEN NEXT

## 6.2.7 Exercise - Summaries

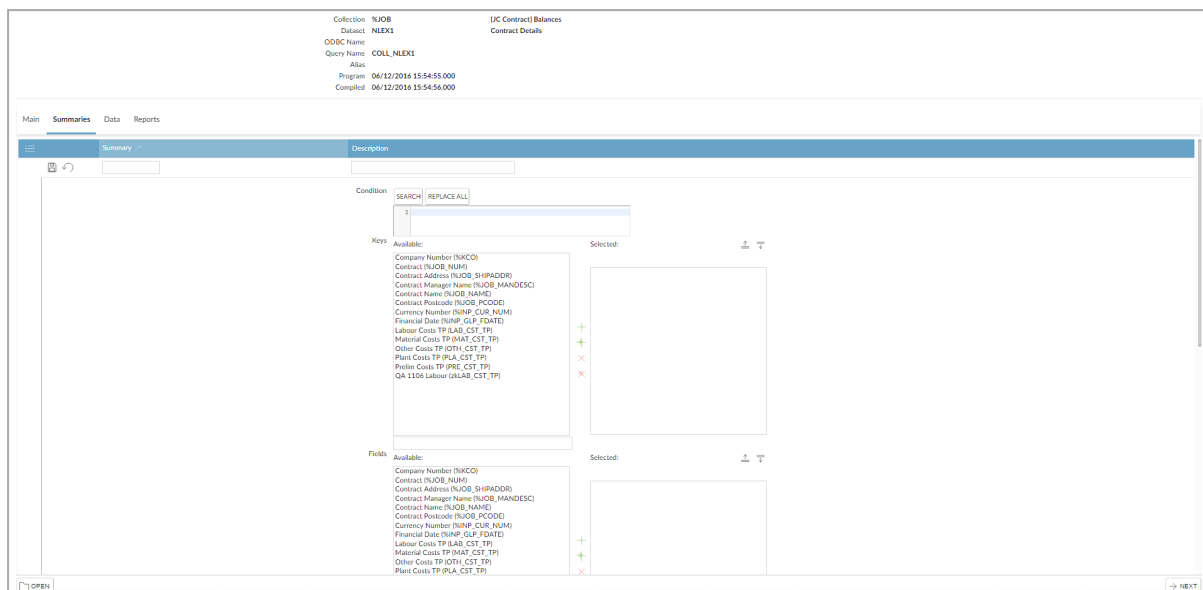
Within the Semantic Layer Dataset, you can have summary information created at the same time as the detailed data. These summaries can then be accessed within queries without the need to build separate datasets. This may be useful for running a series of inline reports based on detail and summary data – the dataset needs only be built once to feed each of the reports.

In the dataset we created in Exercise 1, open the dataset and click the Summaries Tab.



Here we can define different levels of summary, depending on your requirements.

Click



The summary name can be any name you like, but COINS best practice is to use a three letter short code for the separate keys. KCO, JOB, MAN etc. for keys such as Company, Contract and Manager. Thus a summary on Company would be called KCO, a summary based on Company and Manager would be KCOMAN

- Enter KCO as the summary name for our first summary level.

The Keys define which fields the summary will be based upon and the order you define them will set the order of the summary breakdown.

- Select the Key Company Number (%KCO)

The fields' selector determines the fields that you want brought across into the summary. All numeric fields will be included by default and do not need to be selected, but text fields such as Company Name, Account Number etc. will not come across unless you specify them. The fields you bring across should be relevant to the summary keys you set, so in this example we are summarising by company, so contract names and contract numbers will be irrelevant.

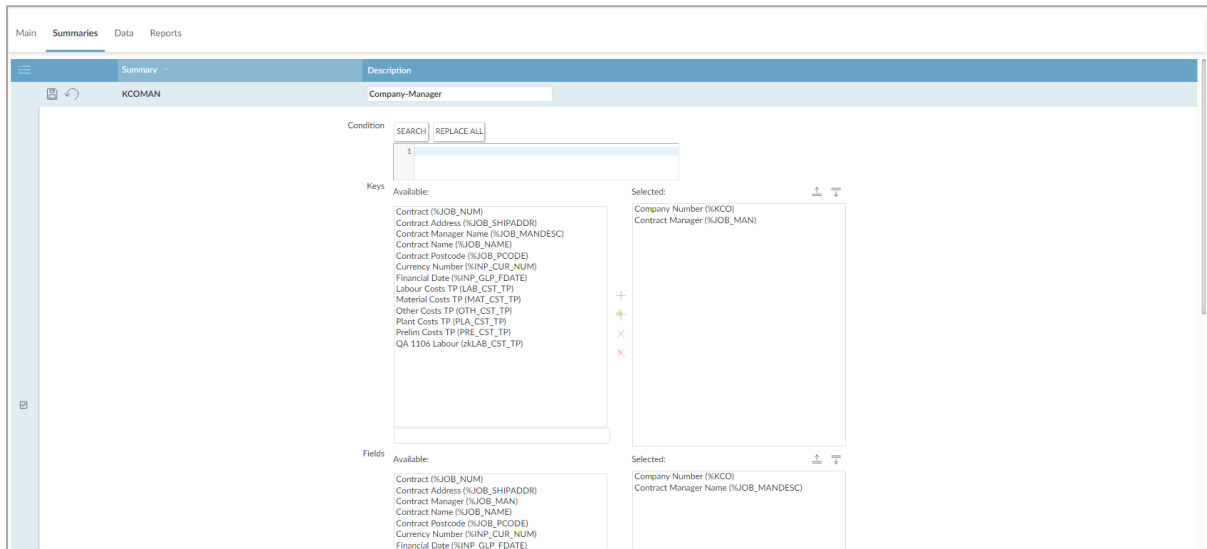
- Select the Company Number field



Use the Summary name KCOMAN and Description Company-Manager

Set the Keys to Company Number (%KCO) and Contract Manager (%JOBMAN)

Set the fields to Company Number (%kCO) and Contract Manager Name (%JOB\_MANDESC)



Click Save and then return to the Data Tab

Re-run the data and scroll to the last set of records





## 6.3 Report Builder

Report Builder is intended for use by end users who are comfortable with using Excel. It is a drag and drop report writing tool which allows the user to lay out banded reports, with groupings, subtotals and totals. It also provides a powerful way to extract data into Excel.

Users can use two types of data source: prebuilt datasets, or custom-built datasets using the semantic layer. Calculated fields and report calculations can do many of the calculations Excel users are familiar with, both on character fields and on values.

A video demonstrating the use of Report Builder is available:

<http://vimeopro.com/coinslms/coins-oa-report-builder-1103>



## 6.3.1 Initial Setup

### 6.3.1.1 Browser Version

Please note that Report Builder has been tested on IE 11 and Chrome. It is NOT available for use with IE 9.

### 6.3.1.2 Report Builder System Parameters

There are two new SY parameters

Parameter ID	Description
RBFUNCS	<p>A comma-separated list of the calculation functions that are available in Report Builder.</p> <p>If this is blank, the default list is: absolute, can-do, date\$, datestring, day, decimal, entry, entry\$, if, if\$, ifexec, ifexec\$, index, inlist, integer, left\$, length, max, min, month, nonzero, numentries, range, right\$, round, string\$, substring\$, sum, today, truncatetrim\$, weekday, year.</p> <p>* means all functions are available.</p>
RBPAGES	<p>A comma-separated list of the report page layouts that are available on a report builder report. For example: %A4RLAND,%A4RPORT,%A3RLAND,%A3RPORT.</p> <p>Blank will show all page layouts.</p>
RBSTYLES	<p>A comma-separated list of the report styles that are available on a report builder report. For example: %8PT,%6PT,%10PT.</p> <p>This allows just a few select styles to be shown rather than all that are defined for other reasons in COINS. Blank will show all styles.</p>

These parameters are not essential to the running of report builder and can be left blank.

#### 6.3.1.3 Report Data Sources – Pre-Built Datasets

The function previously called Report Writer Query has been renamed Report Data Sources and now allows suitable data sources to be flagged as enabled for Report Builder.

Report Builder queries **MUST** use only datasets to extract data; they cannot perform direct queries on the database (which was allowed with Report Writer queries).

Only those data sources selected as Report Builder enabled will be shown in the data sources combo in Report Builder.

#### 6.3.1.4 Security

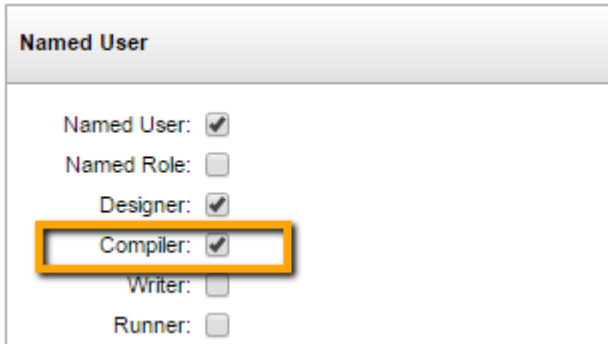
The access that users have in Report Builder to data sources (existing report data sources and collections for custom data sources) relies on function security. This requires new functions to be set up.

- To provide access to a report data source, you must create a function with a function code in the form DSRC\_<data-source-name>. For example, to provide access to data source %JOB, create a function DSRC\_%JOB.
- To provide access to a semantic layer collection, you must create a function with a function code in the form DCOL\_<collection-name>. For example, to provide access to collection %JCT, create a function DCOL\_%JCT.

Then use standard function security to grant access to users.

### 6.3.1.5 Query Results Licence

To use custom data sources, you must have a Progress Query Results licence, and each named user needs to have the Compiler option on their user record ticked.

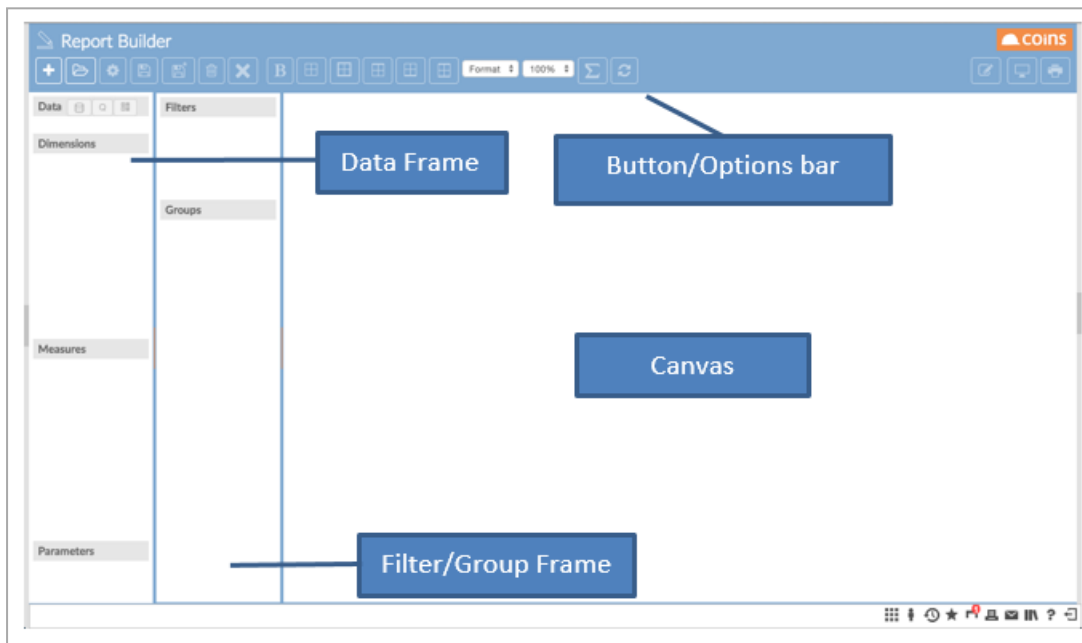
A screenshot of a web form titled "Named User". The form contains several rows, each with a label and a checkbox. The "Compiler" checkbox is checked and highlighted with a yellow border. The other checkboxes are unchecked.

Named User	
Named User:	<input checked="" type="checkbox"/>
Named Role:	<input type="checkbox"/>
Designer:	<input checked="" type="checkbox"/>
Compiler:	<input checked="" type="checkbox"/>
Writer:	<input type="checkbox"/>
Runner:	<input type="checkbox"/>

### 6.3.2 Report Builder

When you first run Report Builder (%WSY1150WRTN), it shows an empty report screen, which consists of:

- Canvas
- Button/options bar
- Data Frame
- Filter/Group Frame



The page is disabled apart from the “New Report” and “Open Existing Report” buttons.





### 6.3.2.1 Canvas

The canvas shows the current report (work in progress – WIP). When you open a report or start a new one it is the WIP report that you are looking at and changing. Each change is saved immediately; you can leave the WIP report, quit the system and log back in and it will still be there. The WIP report can be discarded or overwritten with the report action buttons as described below; it is only when you Save or Save As that the report is saved permanently.








The canvas is disabled if the Report Type is Spreadsheet Only (see Creating a New Report).

### 6.3.2.2 Button and Options Bar

#### Report Actions



These buttons run actions that affect the report as a whole.


-  New Report – discard the current report and create a new one
-  Open Existing Report – discard the current report and open an existing one
-  Properties of Current Report – change the properties of the current report
-  Save Current Report – save the current report with the existing name
-  Save Current Report As – save the current report with a new name. The previous save of the current report will still exist
-  Delete Current Report – discard the current report and delete the saved version
-  Clear Current Report – discard the current report but retain the last saved version

#### Field Formatting



These buttons apply formatting to fields selected on the canvas. Select a field by clicking on it, or select multiple fields using CTRL+click (Windows) or ALT+click (Mac). Then use the buttons to format the field or fields.

 Bold

 No border

 Full border

 Top border



Bottom border



Top and Bottom border

#### Canvas View



The View combo allows you to show the field names, the field labels or the field format on the canvas.

The Size combo allows you to set the scale of the canvas.

#### Actions



These buttons apply actions to the report canvas.



The Sum button allows you to automatically build total forms for the fields in the body. You can select which forms are built. The forms selected will be deleted and rebuilt using the configuration of fields in the body. Any changes that you made to the total forms will be lost.



The Refresh button rebuilds the canvas from the saved report data in COINS.

#### Run



These buttons relate to running the report.



The Designer button takes you to the report designer page for the current report. This is useful for users who have report designer skills and want to see the underlying data that is being created by the report writer.



The Preview button generates a sample report based on up to 100 records from the data source and immediately shows the result in a PDF in a frame. This preview

may not be available until mandatory parameters have been set.

#### TECHNICAL NOTE

Each time a preview is run, the report output log file is written (overwritten) in `$BASE/var/diag/preview.log`. This log is shared between all users but may be useful for debugging.

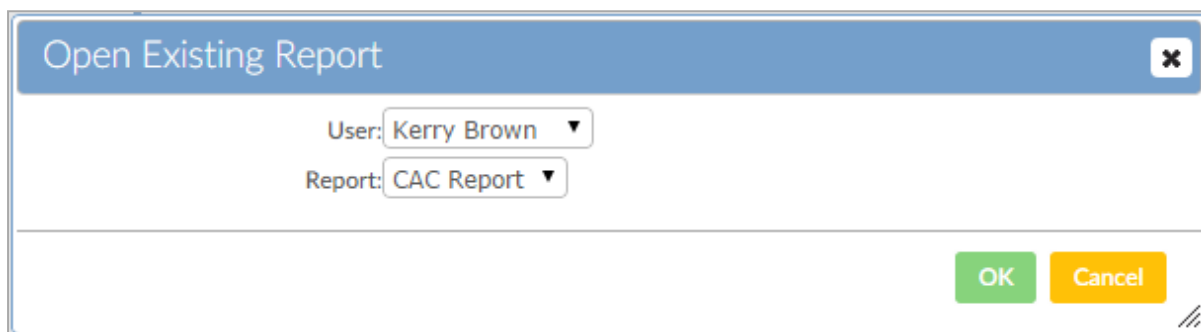


The Report button shows a report selection form (in a dialog) for you to complete and to run the report as any other normal COINS OA report. The resulting PDF and Excel/CSV output will be available on the Report Status Workbench.


### 6.3.2.3 Opening an Existing Report

You can open an existing report created by you (or by another user if you have the appropriate permissions).

Press the Open Existing Report button  This loads the Open Existing Report dialog.

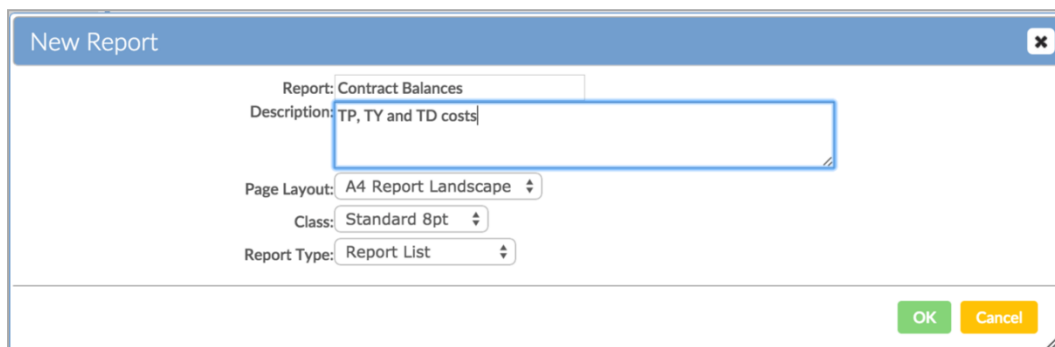


Select the user who created the report and the report you want to open.

To create a new report it may be easiest to open an existing report (which could be another user's report) as above, then use the Save As button  to save it as a new report. This also allows easier report testing – one user can create the report, then another user can copy using this method and test it.

### 6.3.2.4 Creating a New Report

Press the New Report button  to load the new report dialog



#### Fields

Field	Description
<b>Report</b>	The name of the report for you to refer to it. It should be treated like a file name of a document on a disk. Do not use special characters (such as   , " ' ~ \ or ^) in the name.
<b>Description</b>	A fuller description of what the report does.
<b>Page Layout</b>	The page sizes and layout for the report. The list of possible sizes and layouts is controlled by SY parameter RBPAGES.
<b>Class</b>	The font class to use. The list controlled by SY parameter RBSTYLES.
<b>Report Type</b>	The type of report. This can be one of: <ul style="list-style-type: none"> <li>• Report List (normal column listing report)</li> <li>• Report Form (the body of the report is a form rather than a column layout)</li> <li>• Spreadsheet Only (if only a spreadsheet file is required and no PDF is produced). The canvas is disabled with this option; all fields, including calculated fields, will be exported.</li> </ul>

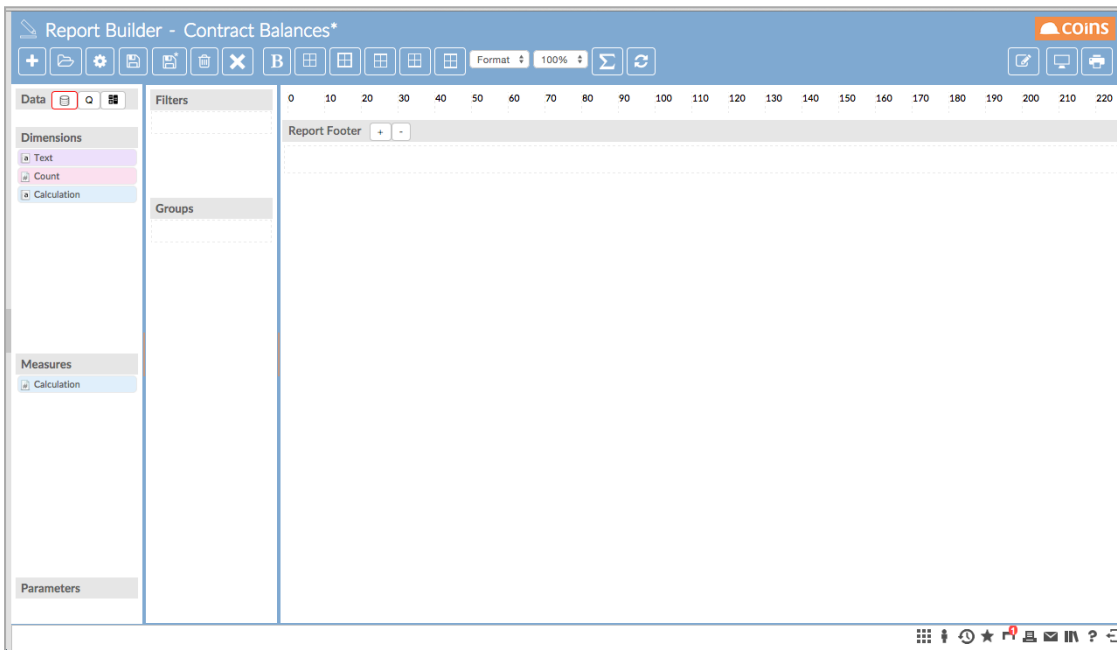
Clicking Cancel would discard the new report definition. Any current report on the canvas would be retained.

When you press the OK button, the busy icon will be shown in the top left hand corner. This is used whenever anything is being sent to the server.

## Report Builder

If necessary, you can change the properties of the report using the  button.

After creating the new report, a blank canvas will be shown. The buttons are now enabled.

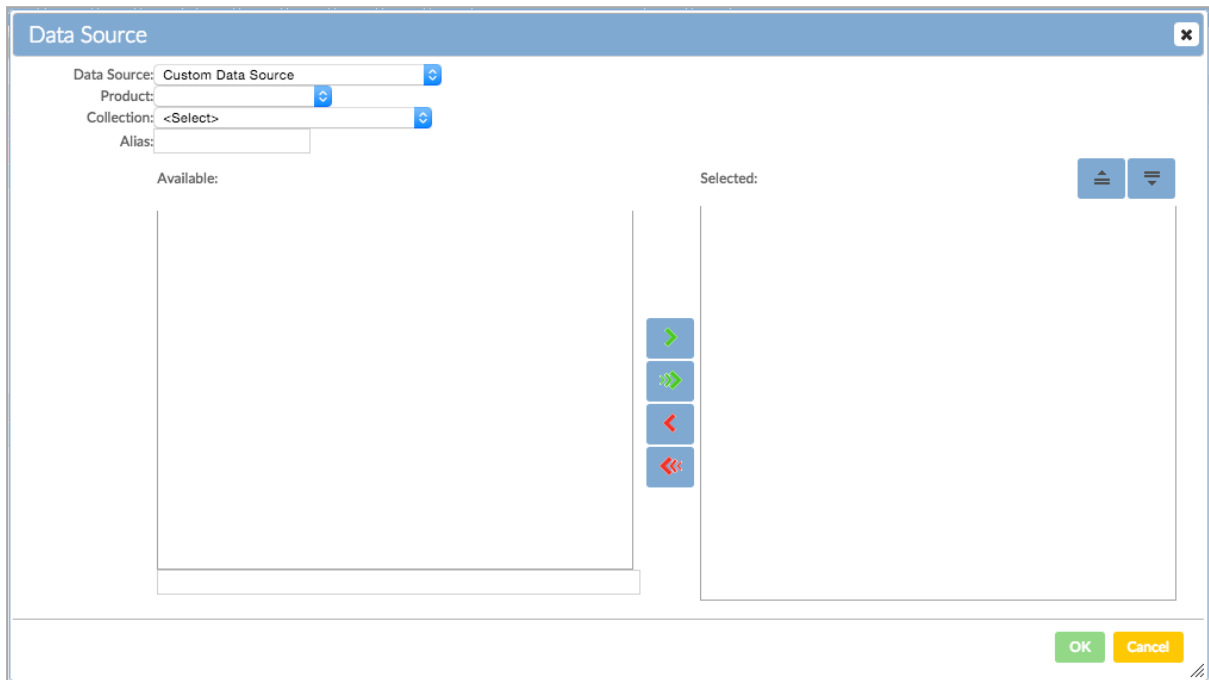


The data source button is highlighted because you have yet to define a data source.



### 6.3.2.5 Data Source

Press the data source button  to display the Data Source dialog.

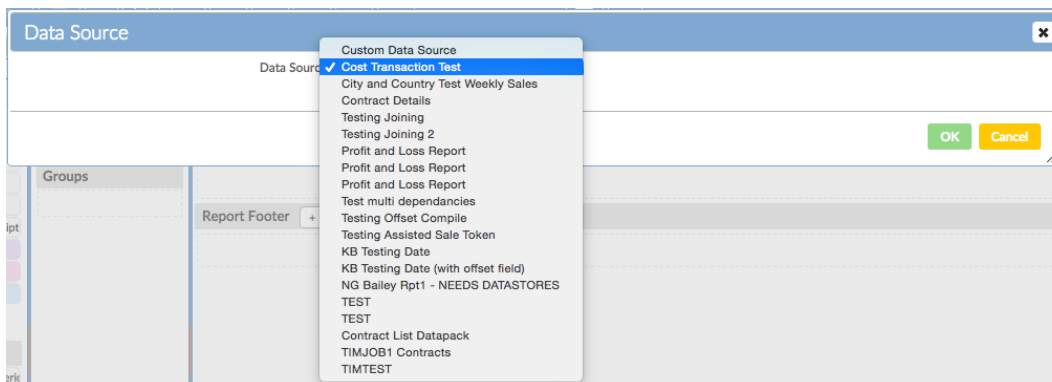


You can select an existing data source or (if you have permission) create a custom data source.

#### Existing Data Source

An existing data source is a data source that has been set up to be available in Report Builder (see 2.3, Report Data Sources).

If you select a pre-defined data source then you simply select from the list of available report data sources.



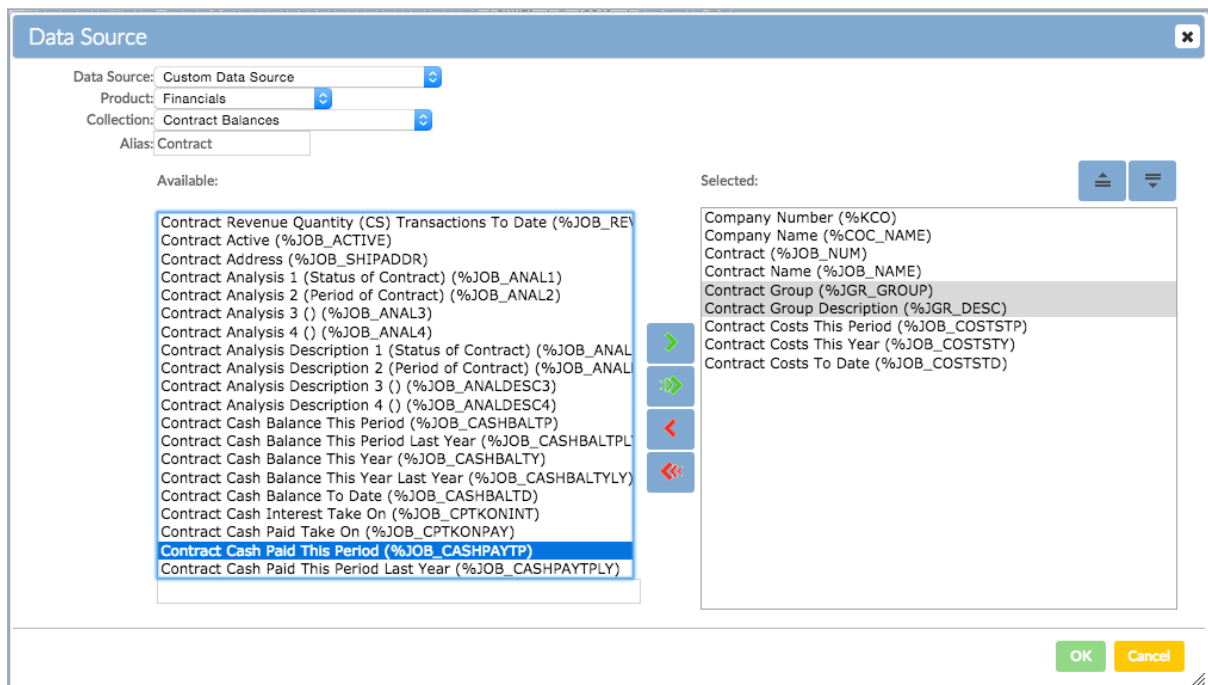


### Custom Data Source

A custom data source is a data set you create 'on the fly' from the semantic layer. You need a compiler licence which allows you to compile data sets from the semantic layer (see 6.3.1.5, *Query Results Licence*).

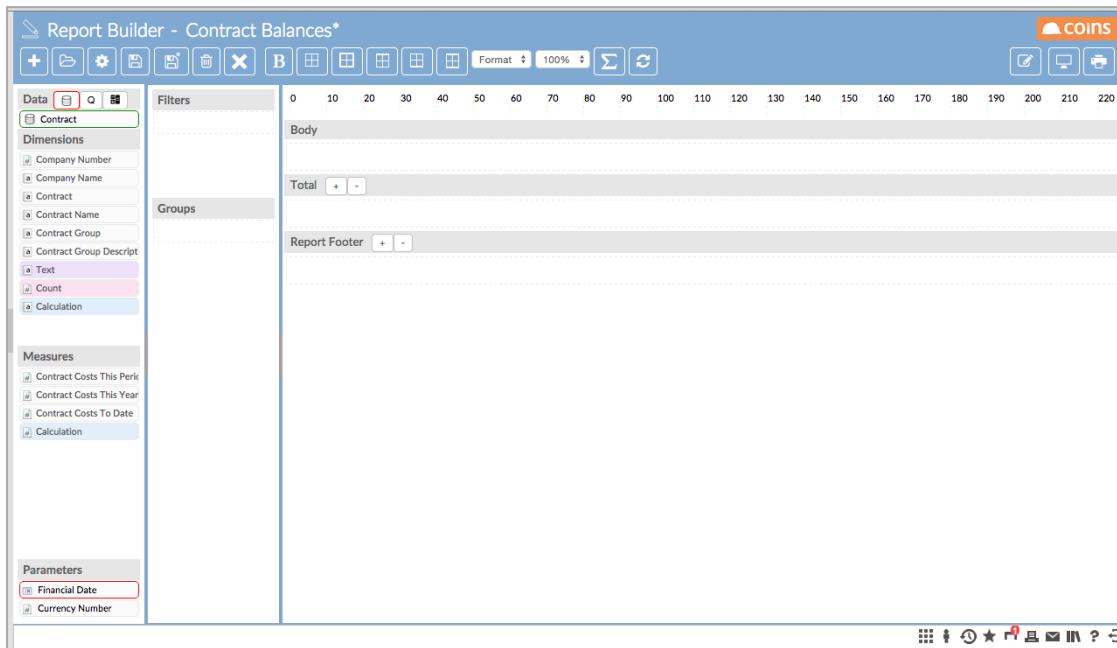
Select a product to filter the list of collections. When you select a collection, the default alias for the resulting table will be copied from the semantic layer and the available fields will be shown.

Select the fields that are required for the report.



Press OK to compile the dataset. You can then define the output.

## Data Frame




The data frame shows:

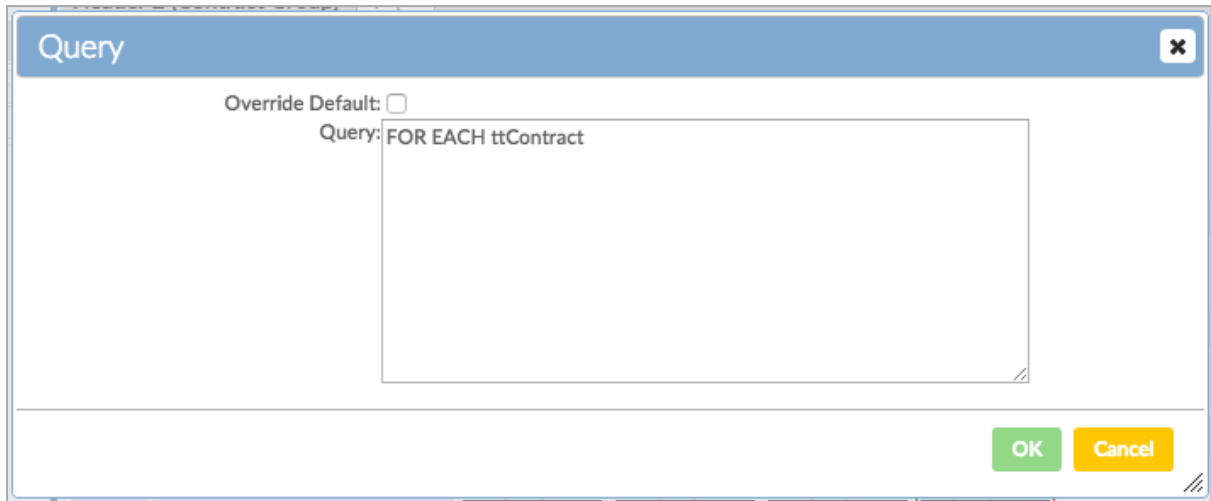
- The tables of the data source (Contract in the example above); there may be multiple tables when using pre-defined data sources. The tables used in the query are shown highlighted in green.
- The fields of the data source, split in to dimensions (values – strings, dates, integers) and measures (decimals – things that can be aggregated).

Dimensions also includes text, count and calculations (for character strings) which you can define when you drag them onto the report canvas (see 3.1.5.3, Text, Count, Calculations). There is a calculation for decimal values in Measures.

- If the data source requires parameters then they are shown at the bottom. Highlighted parameters (Financial Period in the example) are mandatory and yet to be completed.

The canvas has empty body and total forms added to it.

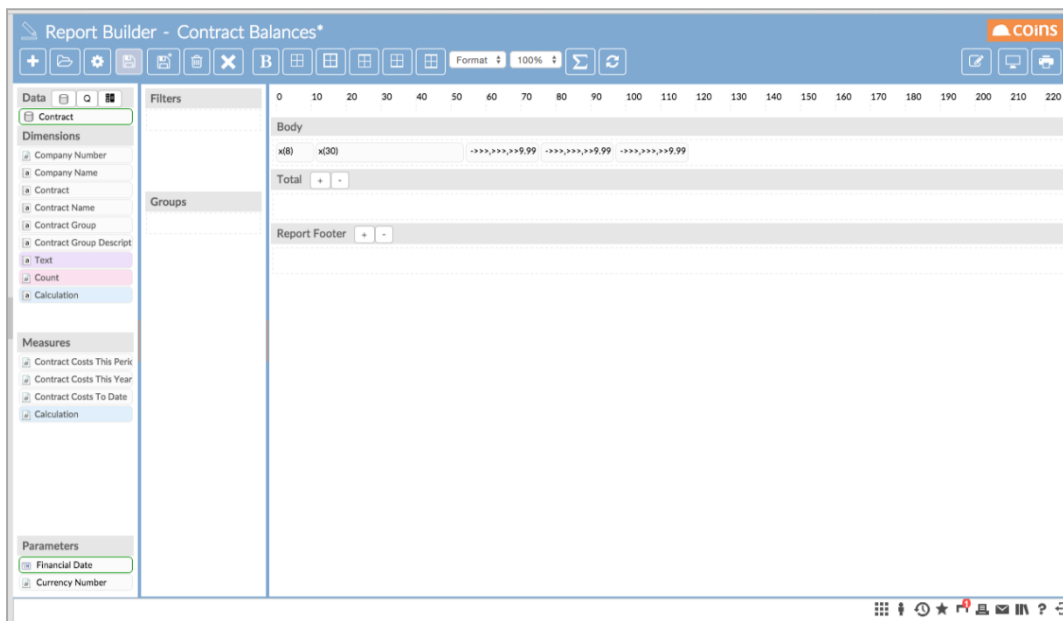
The query associated with the data source is automatically applied. You can view it (and change it if required) by pressing the Query  button.



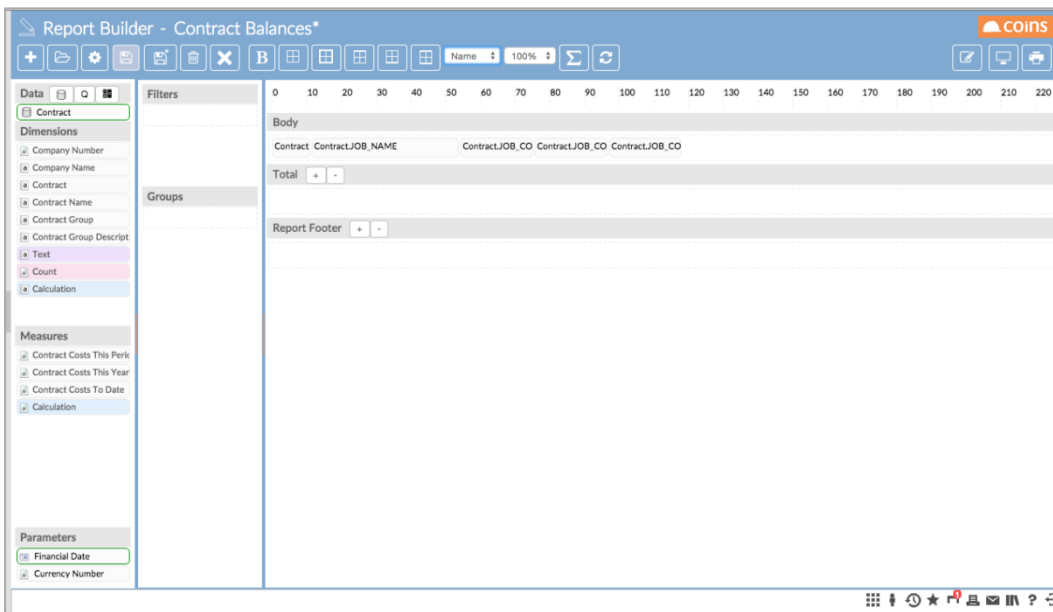
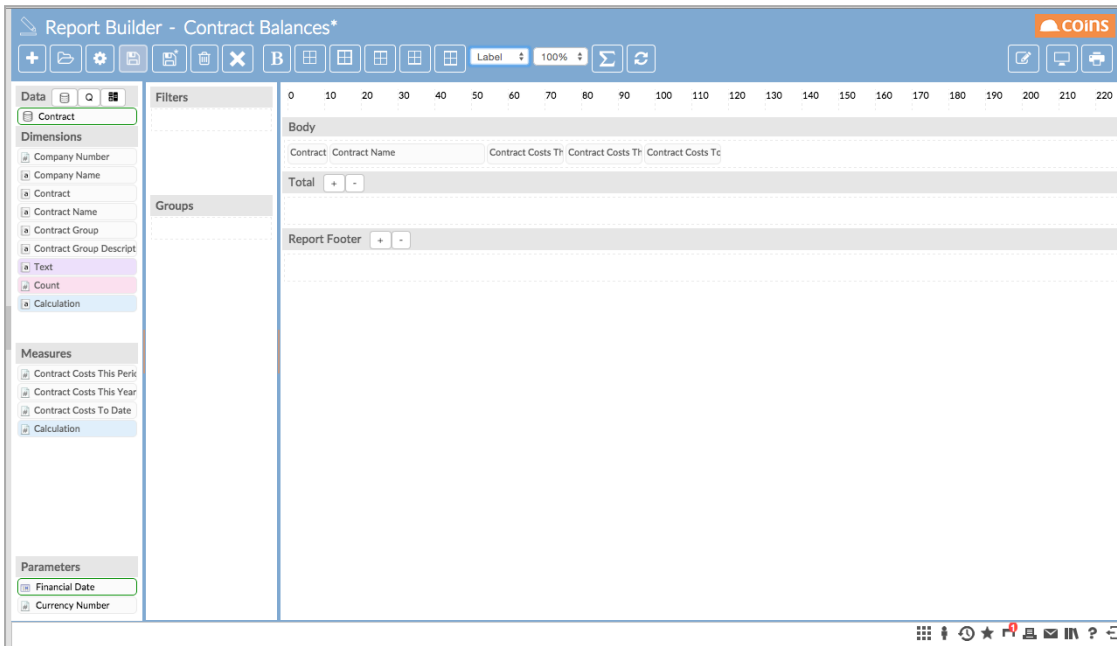
### 6.3.2.6 Adding Fields

To add fields to the report, either:

- Drag a field from the dimensions or measures frame to the canvas and drop it. This allows you to position the fields in the order you want.
- Double-click a field; it will be added to the end of the body form.

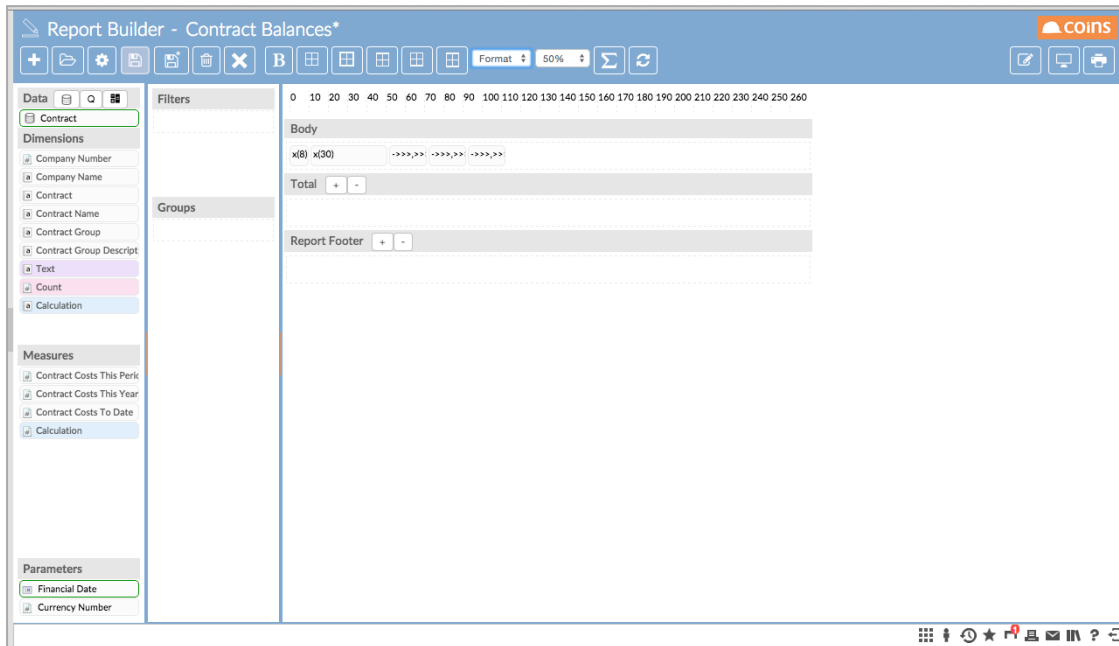



You can change the  show combo in the header shows labels or names.



You can change the scale of the canvas using the Scale selector





 You may notice the Save button pulsing. This indicates that you have made changes to the current WIP report but they have not yet been saved. If you press the Save (or Save As) button, the report will be saved and the pulsing will stop.

Pressing the Preview button  shows the report so far.



Report Builder - Contract Balances

Format 100%

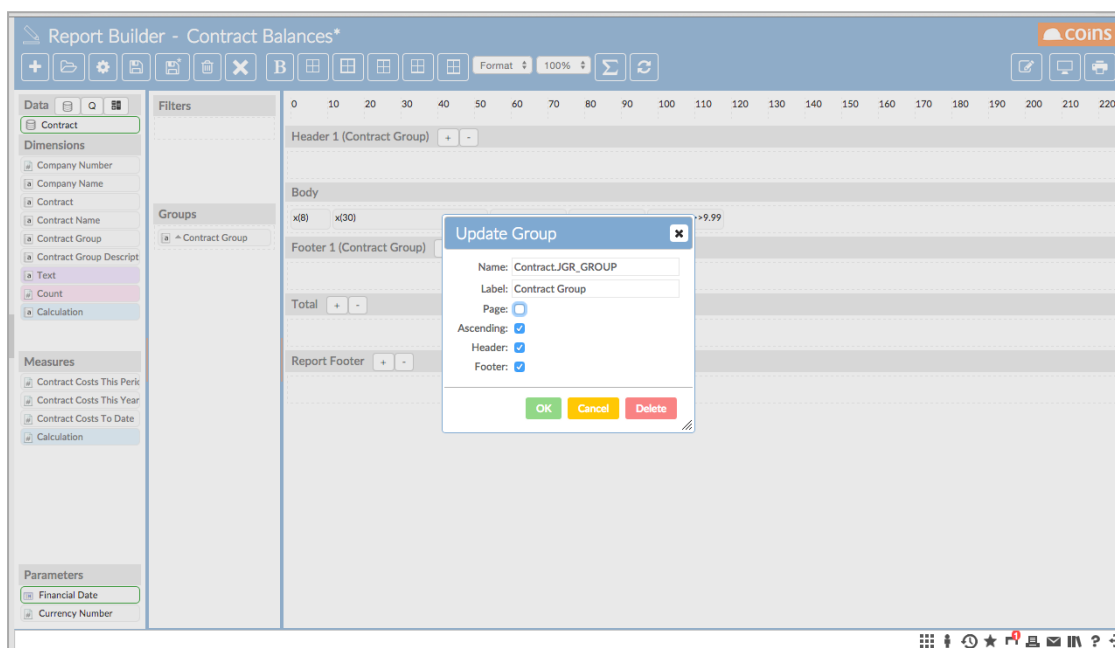
Contract Balances  
CONTRACTOR MASTER

Contract	Contract Name	Contract Costs This Period	Contract Costs This Year	Contract Costs To Date
000006	Antshy contract	-500.00	14,810.00	25,572.50
000012	su-NoWbs NOT change anything here	0.00	103.80	5,478.58
000016	Plant tests, Full Allocation	0.00	0.00	0.00
000017	Plant tests, GRN	0.00	0.00	0.00
000018	Plant tests, Order	0.00	0.00	0.00
1	Aberdeen City Council, No WBS	-8,000.00	64,455.50	66,105.59
1222222	antshy test	0.00	10.00	10.00
123	Contract 123X	0.00	0.00	0.00
1234	test	0.00	27,148.28	30,496.28
1234567890	456	0.00	0.00	0.00
1235	The Wharf (1235)	0.00	-150.00	0.00
1236	1236	0.00	60.00	90.00
23200-1		0.00	0.00	0.00
698	698	0.00	0.00	13.00
777888	ANTFOM F-1 Circle	0.00	0.00	1,245.00
951		0.00	0.00	0.00
A1	A1 Contract	0.00	395.00	1,245.00
A2	A2 Contract	0.00	0.00	0.00
A3	A3 Contract	0.00	0.00	0.00
alebakGR	alebak GRN+WBS	0.00	0.00	-1,985.00
alehal	alehal NO project (WBS)	0.00	0.00	0.00
AleHuG	AleHu GRN+WBS	0.00	0.00	0.00
AleHuO	AleHu Order + WBS	0.00	0.00	0.00
AleHuZ	AleHu WBS (zeror)GRN Plant	0.00	0.00	0.00

Parameters:  
Financial Date  
Currency Number

### 6.3.2.7 Groups

To sort data, and (optionally) create header and footer forms, drag fields from the data frame to the groups frame. You can drag the group fields within the groups frame to change the sort/group order. Double-click a group to show the group dialog.



#### Fields

#### Page

If you tick this, the report will have a new page for each different value of this field.

#### Ascending

If you tick this, the sort order for this field will be ascending. Otherwise it will be descending.

#### Header/Footer

Whether a header and/or footer form for this field should be shown on the canvas. If you are only using the group to control sorting, you may prefer not to show the forms. If a header or footer form is shown on the canvas but is empty then it will not appear on the report.

You can now drag fields to the new forms.



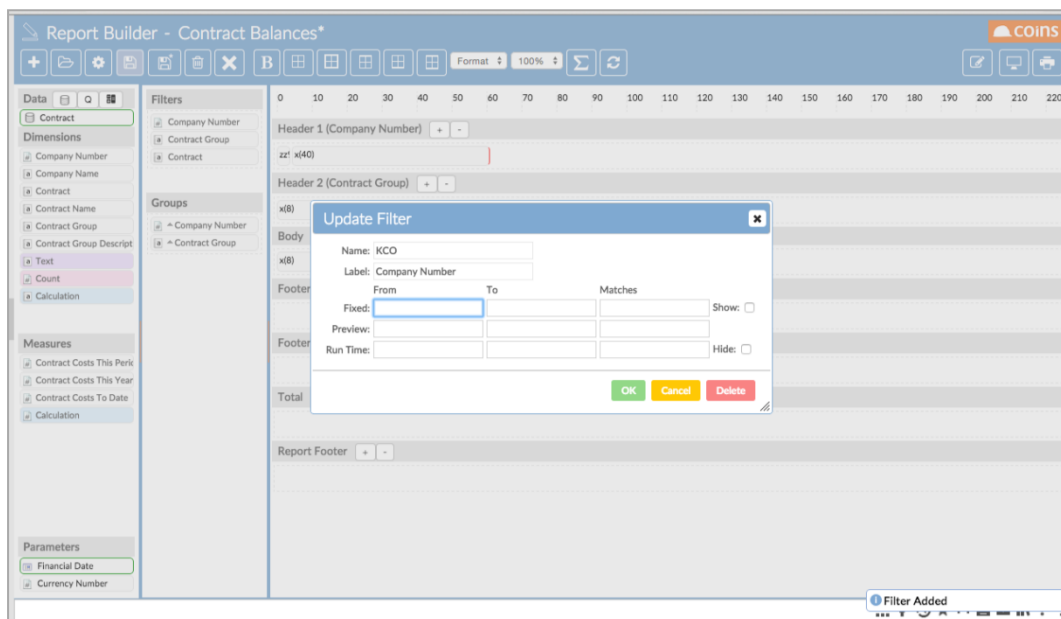


The screenshot displays the 'Report Builder - Contract Balances\*' interface. On the left, there are three main sections: 'Dimensions' with items like Company Number, Company Name, Contract, Contract Name, Contract Group, Contract Group Descript, Text, Count, and Calculation; 'Measures' with items like Contract Costs This Perik, Contract Costs This Year, Contract Costs To Date, and Calculation; and 'Parameters' with Financial Date and Currency Number. The 'Filters' and 'Groups' sections are currently empty. The main report area shows a table structure with columns numbered 0 to 220. The table includes sections for Header 1 (Company Number), Header 2 (Contract Group), Body (with a formula: x(8) x(30) --->>>,>>>9.99 --->>>,>>>9.99 --->>>,>>>9.99), Footer 2 (Contract Group), Footer 1 (Company Number), Total, and Report Footer. A 'Field Saved' notification is visible at the bottom right.

### 6.3.2.8 Filters

To create a filter, drag fields to the filters frame. The order shown is the order they will appear on the input form for the user when running the report. You can re-order the filters by dragging them within the filters frame.

Double-click a filter to show the filter dialog.



The From, To and Matches columns are for the filters and act the same as normal from, to and matches filters in COINS.

#### Fixed

If you tick this, the filter will be fixed (users cannot override it at run time).

For example, you might use this to omit a specific joint venture company by putting !9,\* in the matches column. This would mean not company 9, but all other companies. The user then might still override which of the other companies they want to see at run time but they would never see company 9.

The Show option allows this fixed filter to be shown on the report selection screen (if desired).

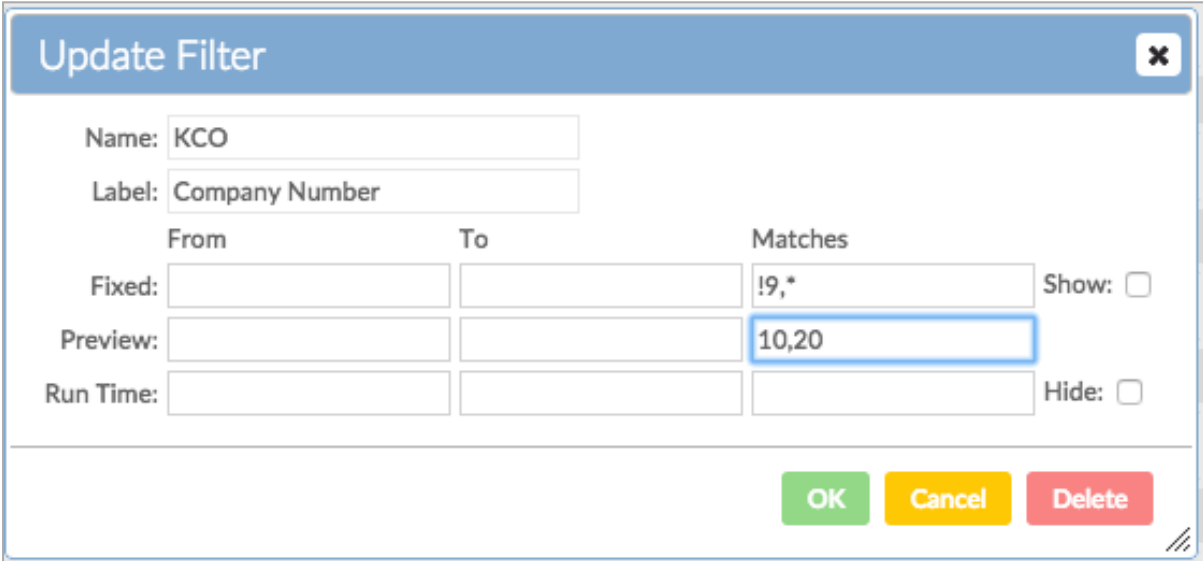
#### Preview

The values that will be used at preview time only. They are not used at run time. This allows you to pick some useful data to be shown on the preview report.

### Run Time

Default values for the report selection screen.

If you do not want to allow the user to change the filter, tick the Hide option (in this case you will need to set a fixed filter).



	From	To	Matches	Show	Hide
Fixed:			!9,*	<input type="checkbox"/>	
Preview:			10,20	<input checked="" type="checkbox"/>	
Run Time:					<input checked="" type="checkbox"/>

Pressing Preview again will show the report so far.




Report Builder - Contract Balances\*

coins

Format 100%

220

**Contract Balances**  
**CONTRACTOR MASTER**



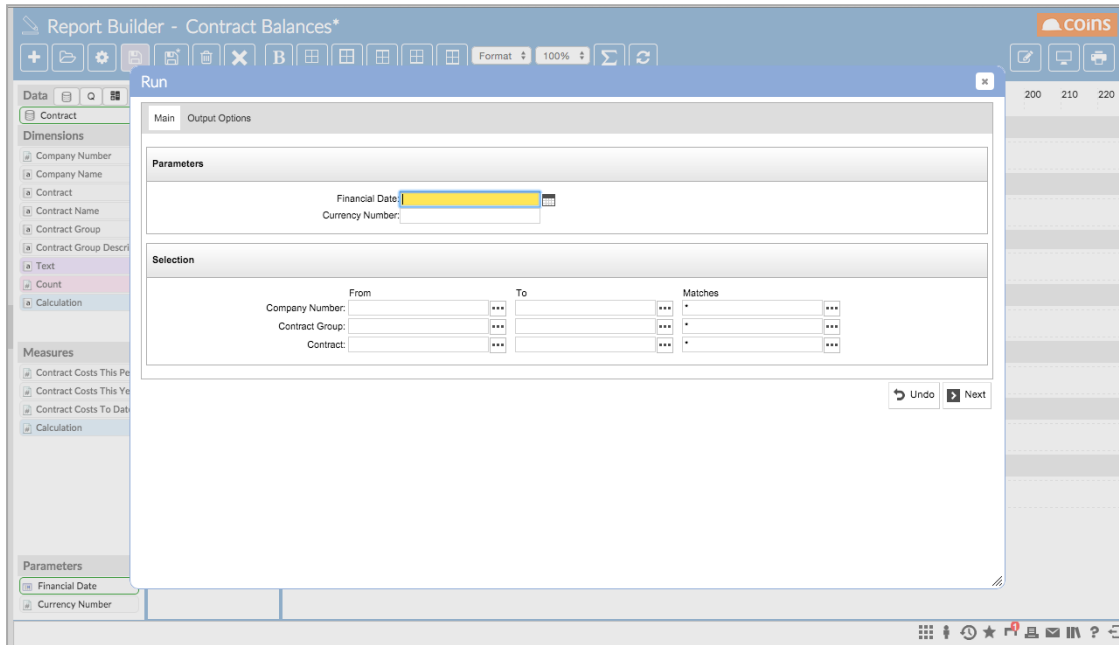
Contract	Contract Name	Contract Costs This Period	Contract Costs This Year	Contract Costs To Date
10	Contractors QA			
00	Sector 00XXxyy1234567			
A1005	North Haverbrook Rail Station	0.00	0.00	0.00
A1004	Ogdenville Town Hall	0.00	0.00	600.00
000020	Aleem Contract Status	0.00	0.00	200.00
000022	Inor Shenyk Test	0.00	0.00	0.00
A1001	Brockway School	0.00	0.00	13,702.00
A1000	North Haverbrook Hospital	0.00	0.00	32,082.79
8001	Irish Contract (no wbs)	0.00	0.00	0.00
8001	The Edge	0.00	0.00	150.00
8000	The Plateau	0.00	0.00	0.00
696	alehui pr 1	0.00	0.00	0.00
1002	Fonseca Estates	0.00	0.00	100.00
1004	Little Aston Park	0.00	0.00	13,419.00
5000	ASDA Slough Electrical Refurb	0.00	-935.00	-815.00
1007	Hamptons Hospital Construction	0.00	50.00	92,413.83
314159	BABYLON TOWER	0.00	0.00	0.00
1009	Test WBS Default v2	0.00	0.00	20.00
108753	Testcontract108753	0.00	0.00	0.00
181285	Hugo Major Contracts	0.00	0.00	-350.00
21122	Winsor House Derby	0.00	0.00	890.00
2012	Jinmao Tower	0.00	0.00	3,630.00
2013	Jinmao Tower	0.00	0.00	0.00
5248S		0.00	0.00	0.00
A1003	Cvoress Creek Librari	0.00	0.00	4,830.00

Parameters:  
 Financial Date  
 Currency Number

The sort/group and (preview) filtering have been applied.

### 6.3.2.9 Run Report

Press the Run Report button  to run the report.



The report selection form is shown in a frame like any other COINS report.

Fixed parameters or filters are shown (as defined) and you can change the run time selection fields. Press the Next button to submit the report to the background queue to be generated. The result will appear on the Report Status Workbench.



001 - CONTRACTOR MASTE | System - Report Status | 1 - CONTRACTOR MASTER - Tim Armitage | COINS

System | Generators Running: | Options

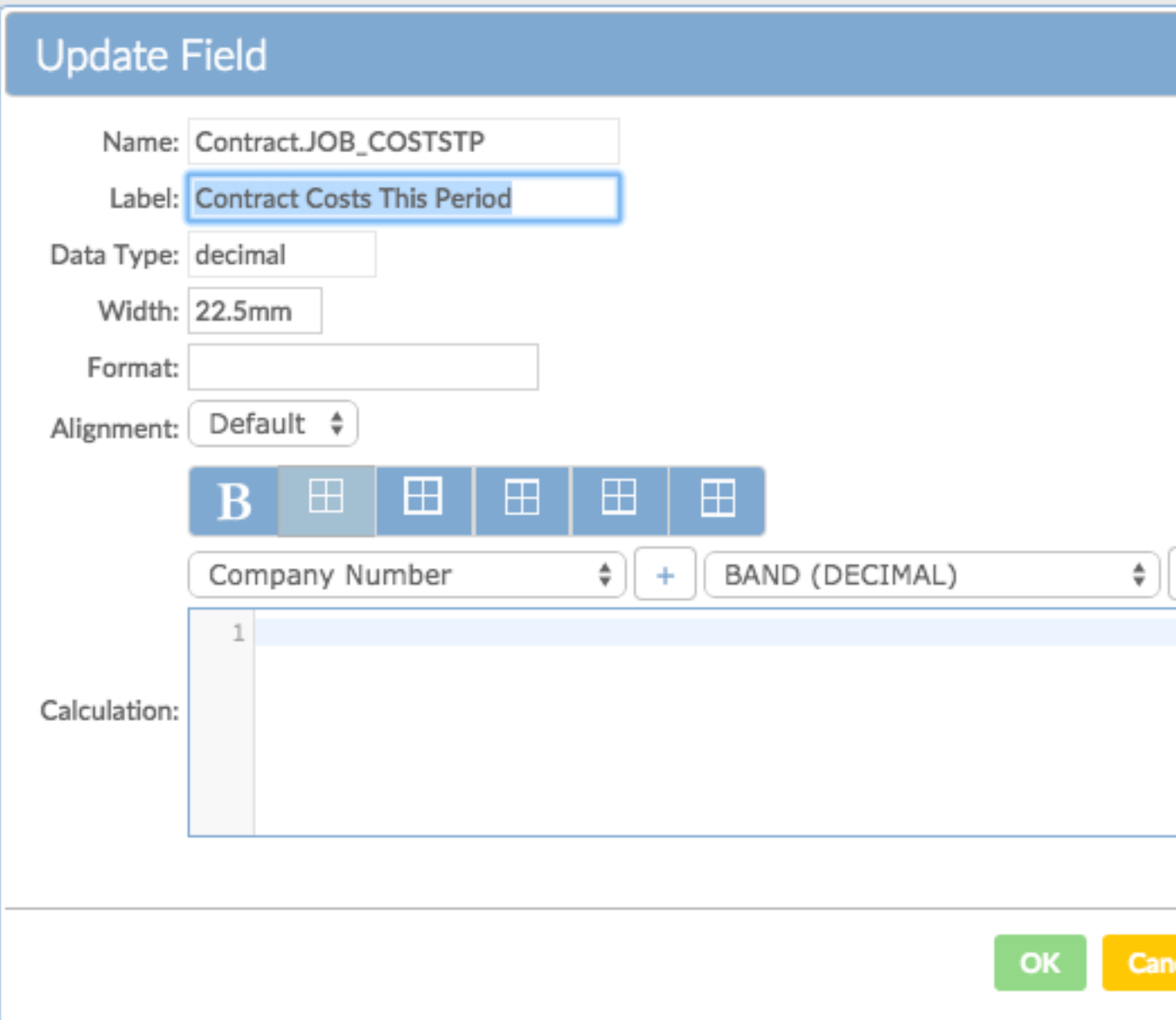
012010 Report Builder | Report Status | Saved Reports | Scheduler | Report Runner | Archived Reports | My Files | Users | Field Sets

	Date	Time	Co	Module	Description	Status	Queue	Size	User			Re-Run
<input type="checkbox"/>	18/05/15	09:54:07 AM	1	SY	<a href="#">Contract Balances</a>	Complete	LOCAL	180,915	TIMARM			

Q\* Advanced | Filter: My Reports | Search: User

### 6.3.2.10 Refining the Report - Fields

Double-click a field to launch the field dialog.



**Update Field**

Name: Contract.JOB\_COSTSTP

Label: Contract Costs This Period

Data Type: decimal

Width: 22.5mm

Format:

Alignment: Default

**B** [Border 1] [Border 2] [Border 3] [Border 4] [Border 5]

Company Number + BAND (DECIMAL)

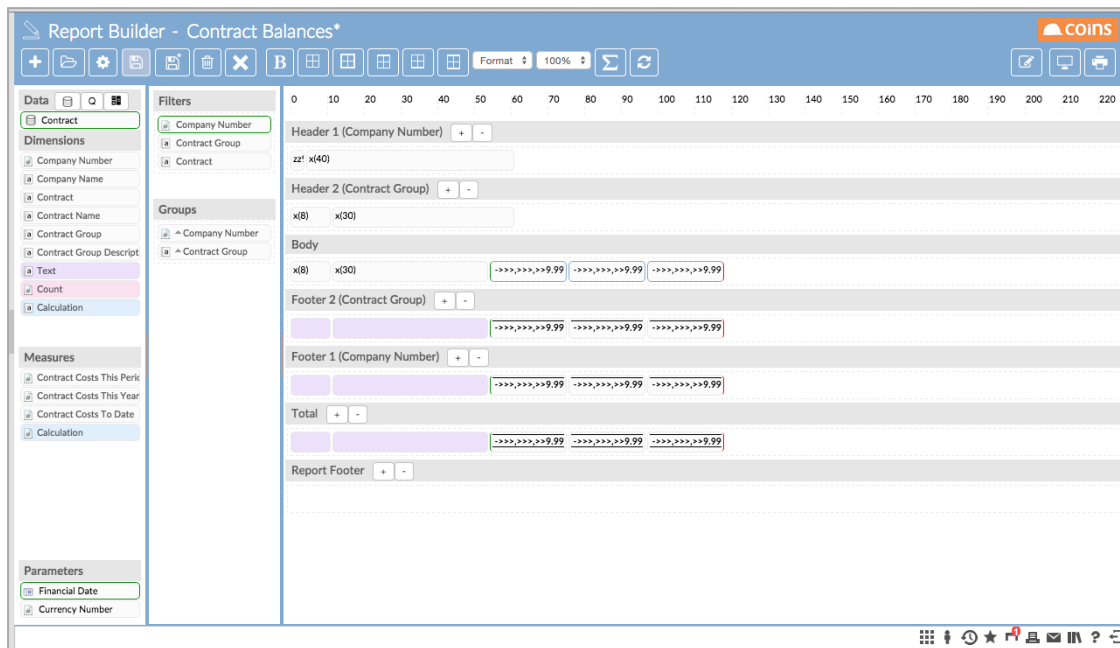
Calculation:

1
---

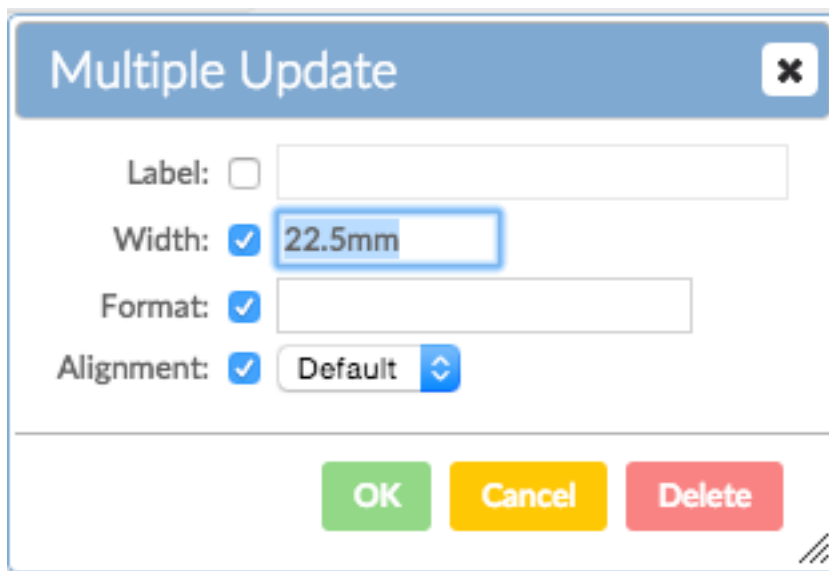
OK Cancel

This allows you to set the label, width, format, alignment, bold and border and specify a calculation. (The bold and border options are also available by selecting one or more fields and pressing the corresponding buttons in the button bar.)

After changing the values in the field dialog, press the OK button to have them reflected on the canvas. You can also press the Delete button to delete the field.



To change more than one field, select one field by clicking on it then use CTRL (or ALT for Mac) to select other fields. They are highlighted with a blue outline. Continue to hold CTRL (or ALT) and double-click one of the fields to bring up the multi field update dialog.



This allows you to set the label, width, format and alignment for all the fields you have selected (similar to multi-update in COINS OA). Similarly the Delete button will delete all the selected fields.



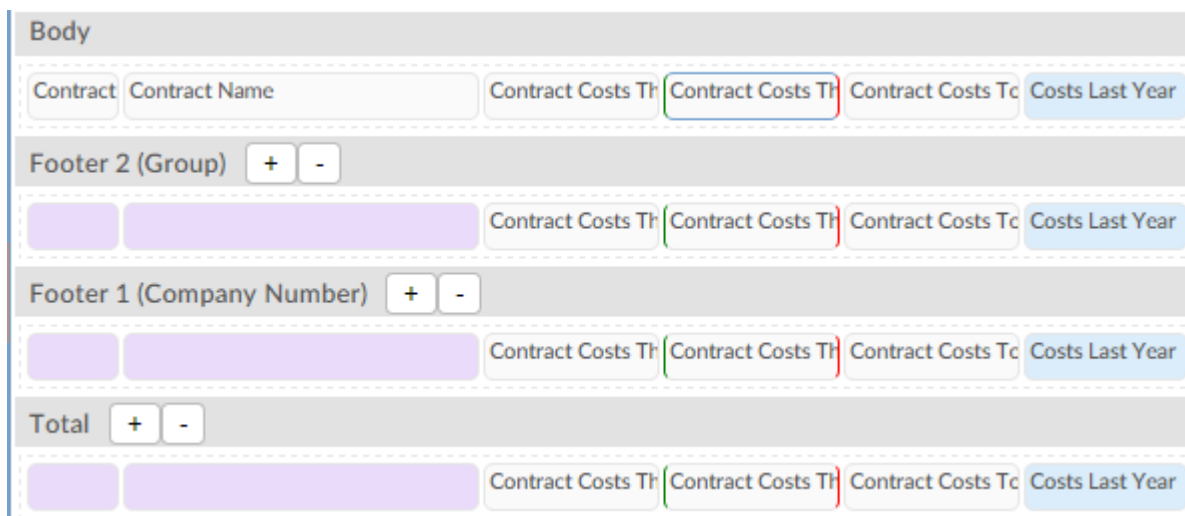
You can also select one or more fields on the canvas and press the Delete key to delete fields. You will be asked to confirm the fields should be deleted.



### 6.3.2.1 Field Positioning

You can drag fields on the canvas to change the order in a form, and you can resize them by dragging the right hand end of the field.

When you select a field (or fields), all fields that match the left side of the selected field (or fields) will be shown with a green left border and all fields that match the right side will be shown with a red right border. This allows you to see when fields are aligned.



When resizing a field, matching fields will show amber right border when close and red when an exact match.

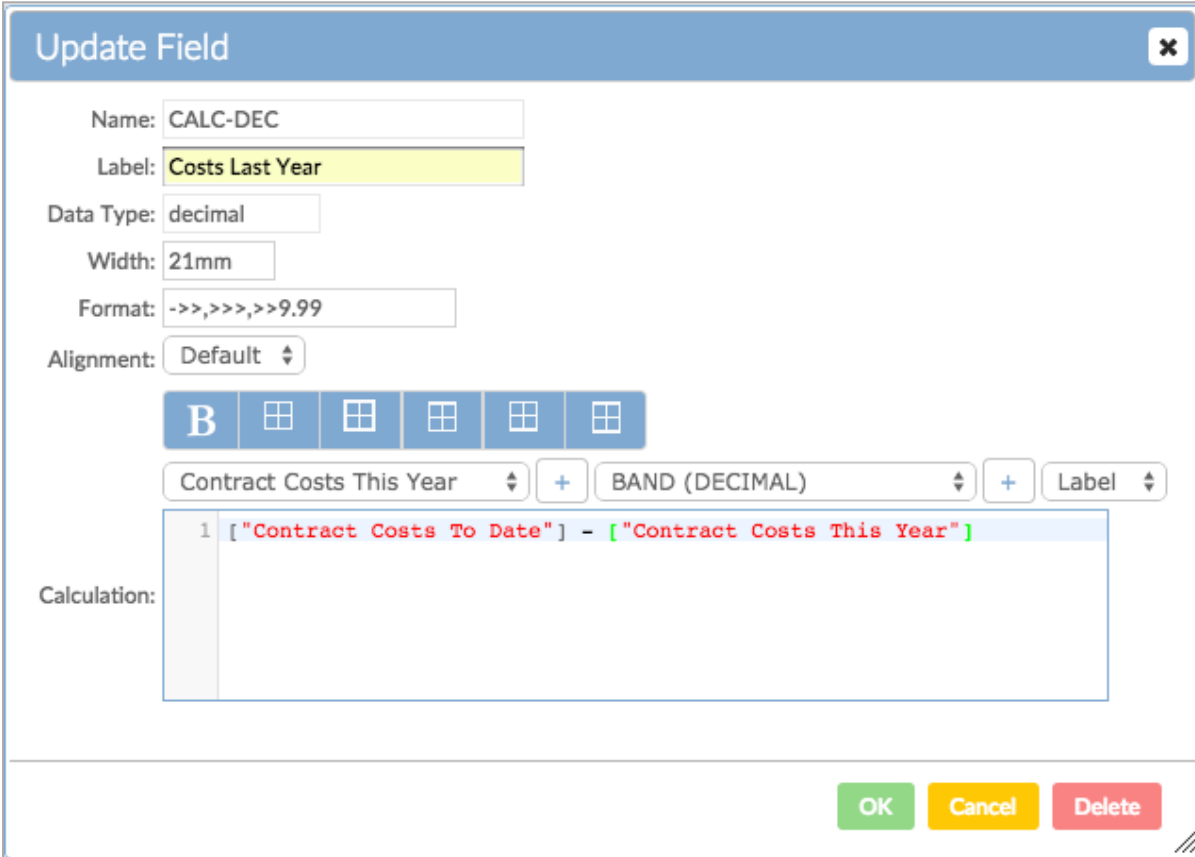
### 6.3.2.1 Text, Count, Calculations

If you drag a text, count or calculation field on to the canvas then you will be able to set its properties by double-clicking like any other field.

For Text fields, the Label is the text that is shown on the report. You can use text fields with no label as padding to position other fields.

Counts can be added to header and footer forms and are evaluated automatically.

Calculations you can specify the calculation to be performed.




The selectors above the calculation area show the fields and functions that are available; you can add these to the calculation by pressing the + button.

The calculation helper allows you to show labels or field names (the image above shows labels).

Calculations on a body form use the values from the data set. Calculations defined on a header or footer use the automatically-aggregated values for the group. If you add a



calculation to the body form then generate automatic totals for the report, the calculation is adjusted for the group forms. For example, if you add a calculation like body.profit /

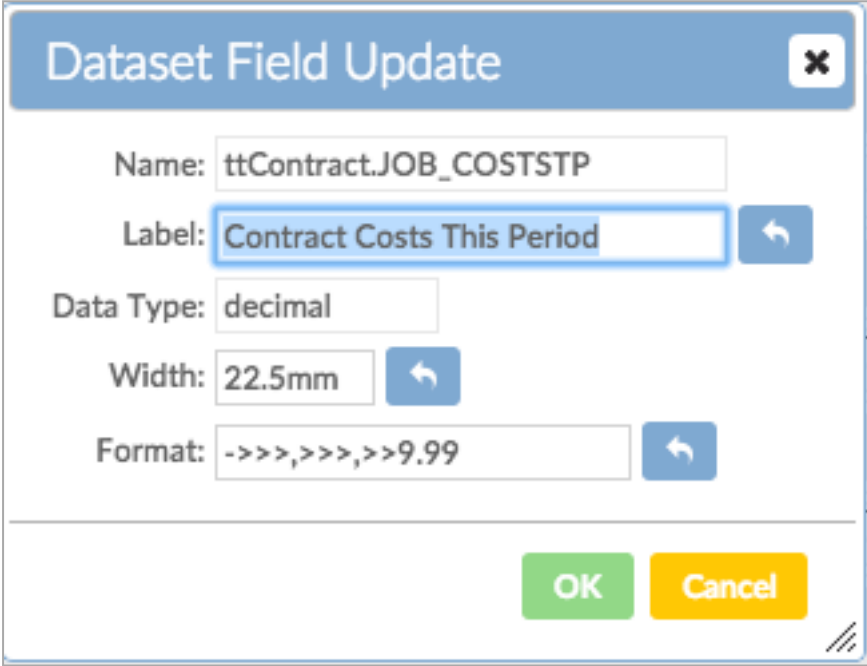
body.costs to the body line, and automatically add it to the group form using the  button, the calculation on the group form will be group.total-profit / group.total-costs.

For more information about calculations, see [Calculations in the BI Toolset](#).

Contract	Contract Name	Contract Costs This Period	Contract Costs This Year	Contract Costs To Date	Costs Last Year
000010	650 Apartments/Office/Leisure	9,616.75	12,152.20	12,421.17	284.97
5588	Apex Offices	0.00	0.00	0.00	0.00
1001	Bovis Test	0.00	60,030.39	91,135.35	31,104.96
10000	Penny Hill Estate	0.00	413.79	88,611.32	58,197.53
000023	Steve Curd Contract 000023	0.00	69,212.22	69,212.22	0.00
21009	Hamptons House	0.00	0.00	0.00	0.00
		<b>9,616.75</b>	<b>142,066.94</b>	<b>233,084.60</b>	<b>91,017.66</b>
BET	Material Betterment	0.00	0.00	311.00	311.00
000014	Aleem	0.00	0.00	311.00	311.00
EL	Sector EL	0.00	0.00	0.00	0.00
1008	New Test	0.00	0.00	0.00	0.00
WEST	West contracts	0.00	0.00	0.00	0.00
22610	Site Stock Contract	0.00	0.00	8,166.67	8,166.67
000622	Jewel of The Creek	0.00	6,050.00	110,738.80	104,688.80
1984	BabyionTower2	0.00	4,000.00	4,093.04	93.04
		<b>0.00</b>	<b>10,050.00</b>	<b>122,998.51</b>	<b>112,948.51</b>
ZZ	Sector ZZ	94,147.99	466,016.37	886,297.22	522,286.85
000150	Ohio Center	94,147.99	466,016.37	886,297.22	522,286.85
		<b>103,764.74</b>	<b>617,248.31</b>	<b>6,564,663.35</b>	<b>5,847,415.04</b>
		<b>103,764.74</b>	<b>617,248.31</b>	<b>6,564,663.35</b>	<b>5,847,415.04</b>

### 6.3.2.1 Dataset Field Update

You can change the default settings on a data source field (for this report) by pressing CTRL (Alt for Mac) and double-clicking the field in the data frame. This brings up the Dataset Field Update dialog.



The screenshot shows a dialog box titled "Dataset Field Update". It contains the following fields and values:

- Name: ttContract.JOB\_COSTSTP
- Label: Contract Costs This Period
- Data Type: decimal
- Width: 22.5mm
- Format: ->>>, >>>, >>>9.99

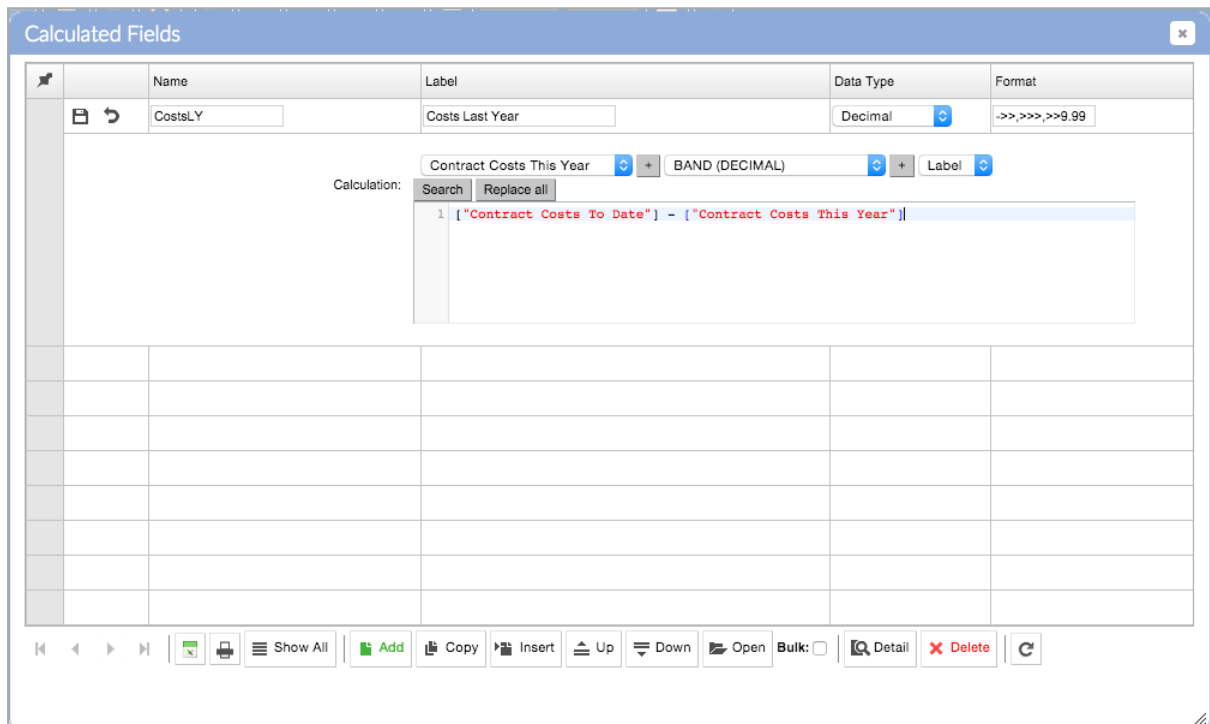
At the bottom of the dialog are "OK" and "Cancel" buttons.

You can override the label, width and format. These will then be used as defaults if you drag the field on to the canvas. The original data source is not affected by these amendments but if the report is copied then they will be retained.

### 6.3.2.14 Adding Fields to Datasets

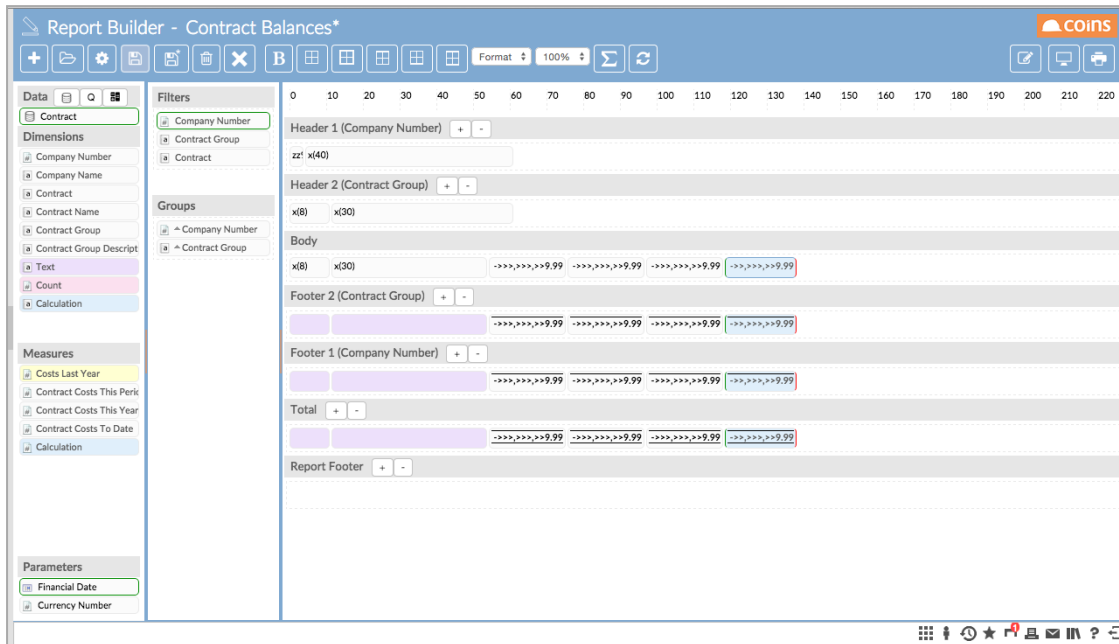
You can add fields to the dataset as part of the report.

Pressing the Calculated Field button  allows you to add calculated fields to the dataset.

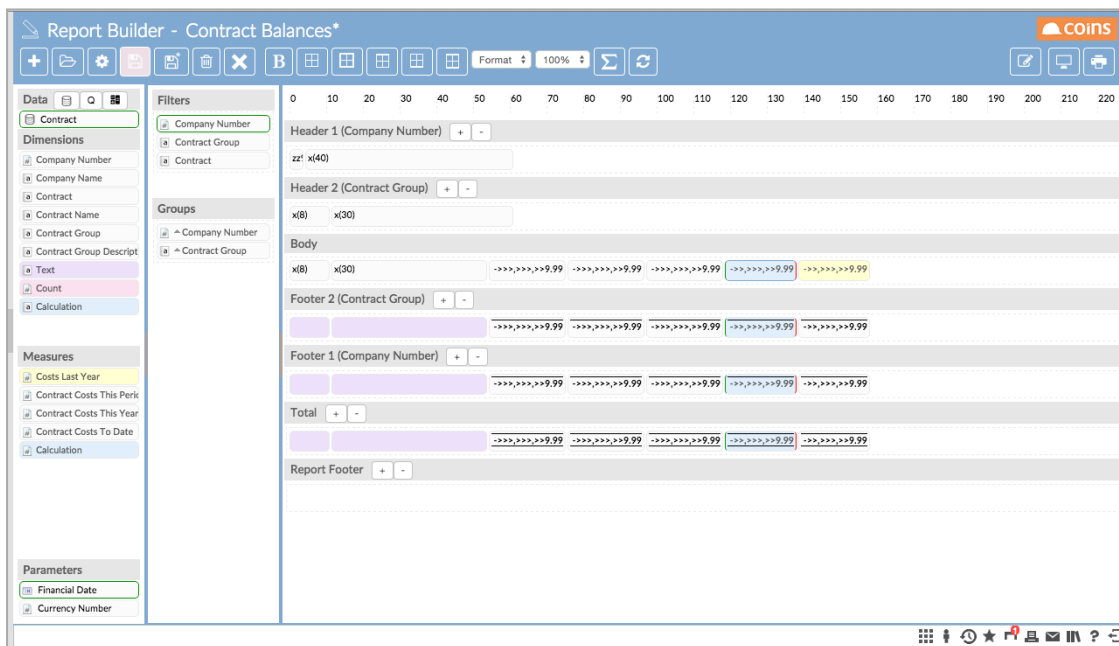


Enter a field name (this must be unique within the dataset), a label, data type and format and the calculation to evaluate to populate the field with data.

When you close the calculated fields dialog, the dimensions and measures will be updated with any calculated fields shown in yellow.



They can now be added to the report like any other field (the example above duplicates the calculation shown in 3.1.5.3).





Report Builder - Contract Balances\*

Format: 100%

220

**Contract Balances**  
**CONTRACTOR MASTER**

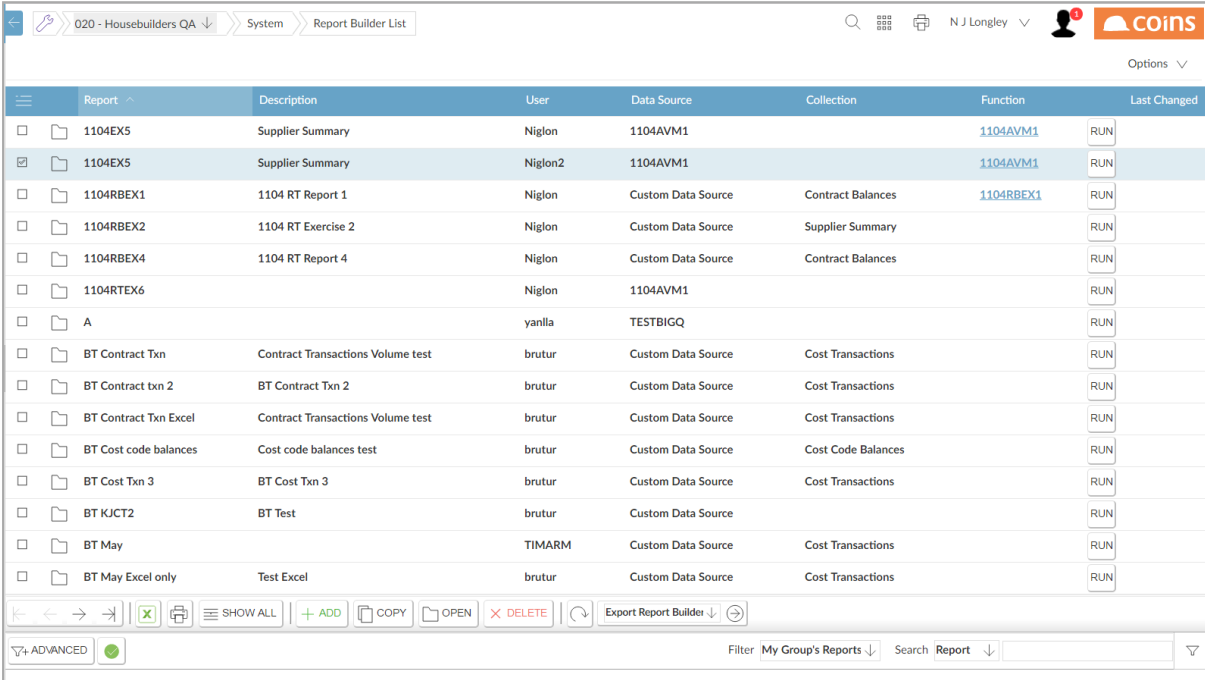
Contract	Contract Name	Contract Costs			Costs Last	
		This Period	This Year	To Date	Year	Year
000010	650 Apartments/Office/Leisure	0.00	12,152.20	12,401.17	0.00	298.97
5588	Apex Offices	0.00	0.00	0.00	0.00	0.00
1001	Bovis Test	0.00	60,030.39	91,135.35	31,104.96	31,104.96
10000	Penny Hill Estate	0.00	413.79	58,611.32	58,197.53	58,197.53
000023	Steve Curd Contract 000023	0.00	69,212.22	69,212.22	0.00	0.00
21009	Hamptons House	0.00	0.00	0.00	0.00	0.00
		<b>9,616.75</b>	<b>142,066.94</b>	<b>233,084.60</b>	<b>91,017.66</b>	<b>91,017.66</b>
BET	Material Betterment					
000014	Aleem	0.00	0.00	311.00	311.00	311.00
		<b>0.00</b>	<b>0.00</b>	<b>311.00</b>	<b>311.00</b>	<b>311.00</b>
EL	Sector EL					
1008	New Test	0.00	0.00	0.00	0.00	0.00
		<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>
WEST	West contracts					
22610	Site Stock Contract	0.00	0.00	8,166.67	8,166.67	8,166.67
000622	Jewel of The Creek	0.00	6,050.00	110,738.80	104,688.80	104,688.80
1984	BabylonTower2	0.00	4,000.00	4,093.04	93.04	93.04
		<b>0.00</b>	<b>10,050.00</b>	<b>122,998.51</b>	<b>112,948.51</b>	<b>112,948.51</b>
ZZ	Sector ZZ					
000150	Otha Center	94,147.99	466,016.37	988,297.22	522,280.85	522,280.85
		<b>94,147.99</b>	<b>466,016.37</b>	<b>988,297.22</b>	<b>522,280.85</b>	<b>522,280.85</b>
		<b>103,764.74</b>	<b>617,248.31</b>	<b>6,564,663.35</b>	<b>5,847,415.04</b>	<b>5,847,415.04</b>
		<b>103,764.74</b>	<b>617,248.31</b>	<b>6,564,663.35</b>	<b>5,847,415.04</b>	<b>5,847,415.04</b>

Param: Financial Date, Currency Number



### 6.3.2.15 Report Builder List

Report Builder List is a new function that allows you to look at all your reports (and if you have the appropriate group permissions, those of other users in your group).



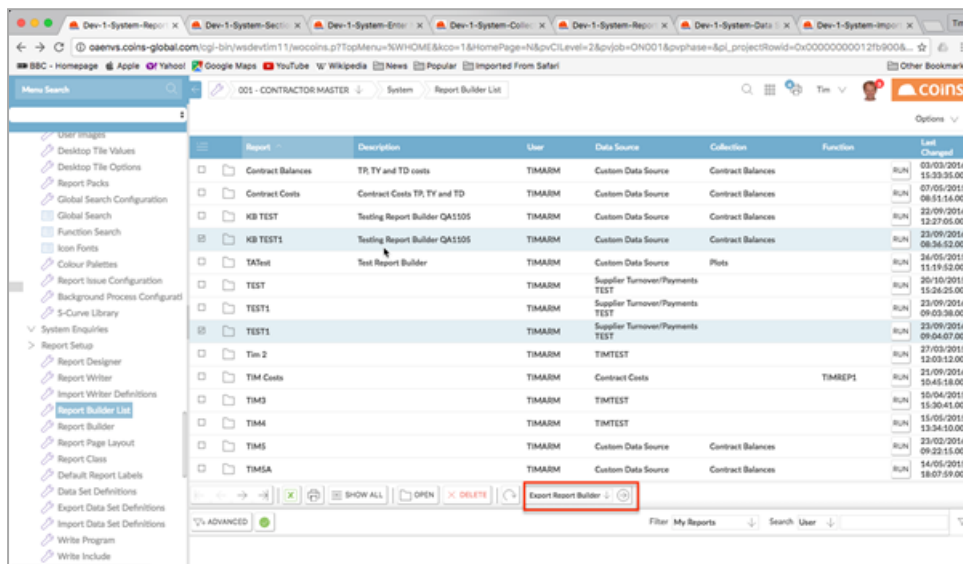
Report	Description	User	Data Source	Collection	Function	Last Changed
<input type="checkbox"/> 1104EX5	Supplier Summary	Niglon	1104AVM1		<a href="#">1104AVM1</a>	RUN
<input checked="" type="checkbox"/> 1104EX5	Supplier Summary	Niglon2	1104AVM1		<a href="#">1104AVM1</a>	RUN
<input type="checkbox"/> 1104RBEX1	1104 RT Report 1	Niglon	Custom Data Source	Contract Balances	<a href="#">1104RBEX1</a>	RUN
<input type="checkbox"/> 1104RBEX2	1104 RT Exercise 2	Niglon	Custom Data Source	Supplier Summary		RUN
<input type="checkbox"/> 1104RBEX4	1104 RT Report 4	Niglon	Custom Data Source	Contract Balances		RUN
<input type="checkbox"/> 1104RTEX6		Niglon	1104AVM1			RUN
<input type="checkbox"/> A		yanilla	TESTBIGQ			RUN
<input type="checkbox"/> BT Contract Txn	Contract Transactions Volume test	brutur	Custom Data Source	Cost Transactions		RUN
<input type="checkbox"/> BT Contract txn 2	BT Contract Txn 2	brutur	Custom Data Source	Cost Transactions		RUN
<input type="checkbox"/> BT Contract Txn Excel	Contract Transactions Volume test	brutur	Custom Data Source	Cost Transactions		RUN
<input type="checkbox"/> BT Cost code balances	Cost code balances test	brutur	Custom Data Source	Cost Code Balances		RUN
<input type="checkbox"/> BT Cost Txn 3	BT Cost Txn 3	brutur	Custom Data Source	Cost Transactions		RUN
<input type="checkbox"/> BT KJCT2	BT Test	brutur	Custom Data Source			RUN
<input type="checkbox"/> BT May		TIMARM	Custom Data Source	Cost Transactions		RUN
<input type="checkbox"/> BT May Excel only	Test Excel	brutur	Custom Data Source	Cost Transactions		RUN

You can run reports directly from this workbench or delete reports that are no longer required. (Deleting is possible in the Report Builder but only one at a time.) Here multiple reports can be deleted at the same time.

You can also update the report record and give the report a function code. This will create a normal COINS function (you will need to give is a suitable function code that is unique) that can be put on a menu for other users to run the report. Once you have given it a function code you will need to grant access permissions to the function and add it to a suitable menu for users to access.

### Export Report Builder Definition

There is a browse action on the report builder list to export the report definitions and the corresponding data sources and data sets.



Select the reports that you want to export and select the Export Report Builder action from the drop down list.

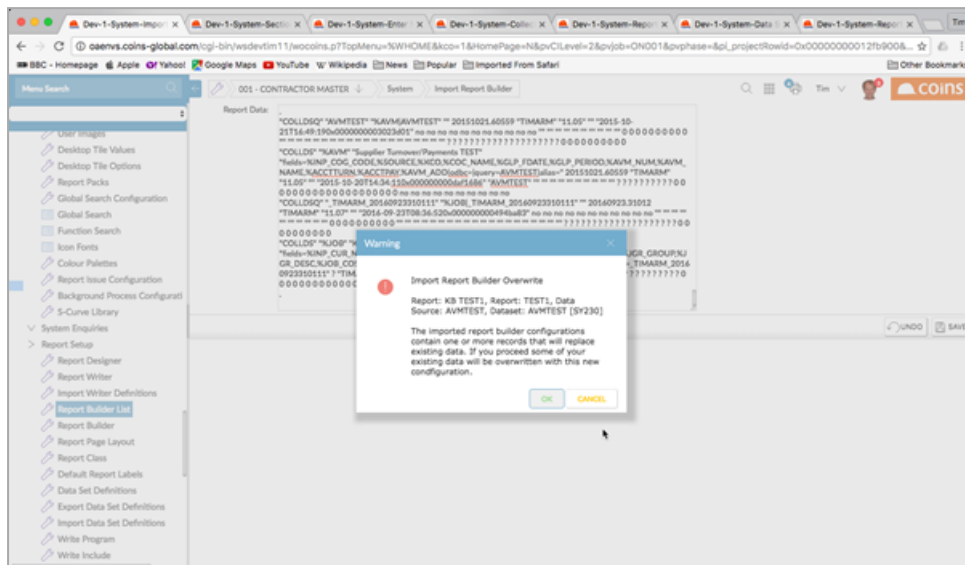
The next screen will show the list of reports to be exported and the list of data sources that will be exported with it.

In the example below, the first of the reports uses a AVMTTEST data source so it is listed. The second report uses a custom data source specific to the report so it is not listed in the data sources.

The report data fields is the exported data that you can copy ready to paste in to the import function.







If you confirm this then report section, form, fields, calculations, data source, dataset and fields, and collection dataset will be imported as required.

You can check that the data source, dataset and collection dataset have been created by looking in the corresponding maintenance pages for these items.

You will be able to load and run the newly imported report(s).

## 6.4 Dynamic Queries

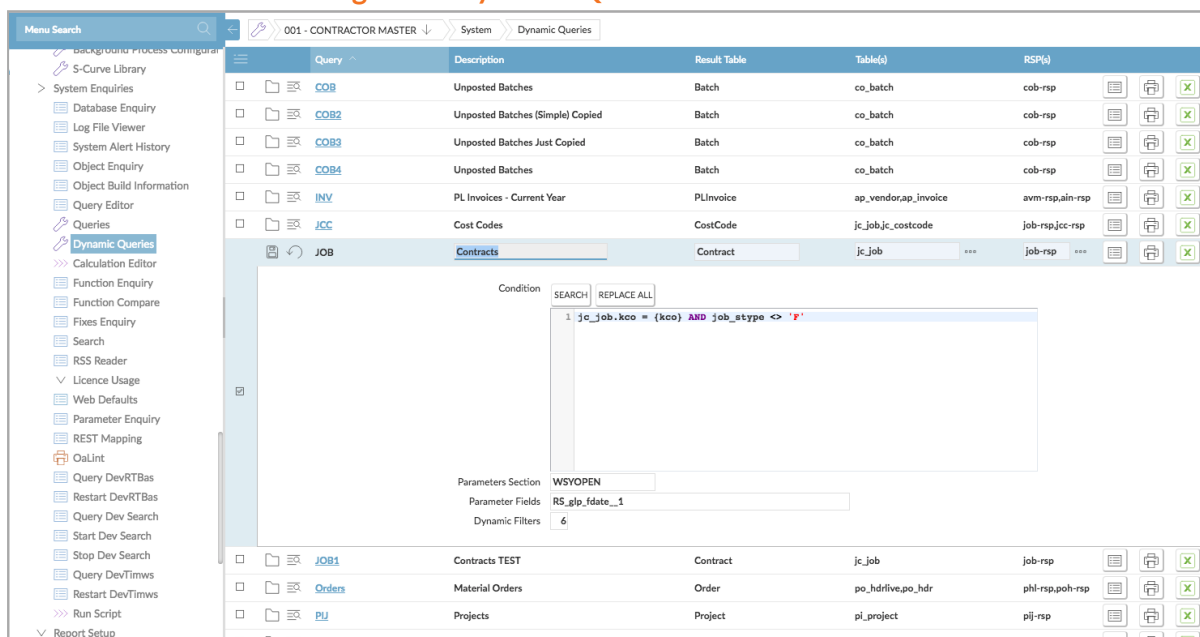
System administrators can now define dynamic queries that can then be made available to users to design their own extracts and grouping of data. There is a new dataset builder and corresponding enquiries, reports and extracts that can use it.

The queries can also be used in the Report Builder.

## 6.4.1 Dynamic Query Definition

**Dynamic Queries** is a new function that allows an administrator to define the dynamic queries (datasets) so that they contain the records and fields that the users require.

**Figure 9: Dynamic Queries Browse Screen**



### Fields

#### Query

The code used to identify the query.

#### Description

A description of the query.

#### Result Table

The temp table name of the resulting dataset.

#### Table(s)

One or more tables that are used in the query.

#### RSP(s)

The corresponding RSPs for each of the tables in the **Table(s)** list. This allows specific RSPs to be used for specific tables.

## Query

This is either:

- A partial query condition that will be appended to each table in the query. One line for each table specified in the **Tables(s)** field, in the same order. The condition should NOT contain the WHERE clause.
- A full query for all the tables specified.

## Parameters Section

If the query requires one or more parameters (for example, a financial date, a currency code or a contract), create a page section update form that will be used to prompt the user for this, and enter the page section code here.

## Parameter Fields

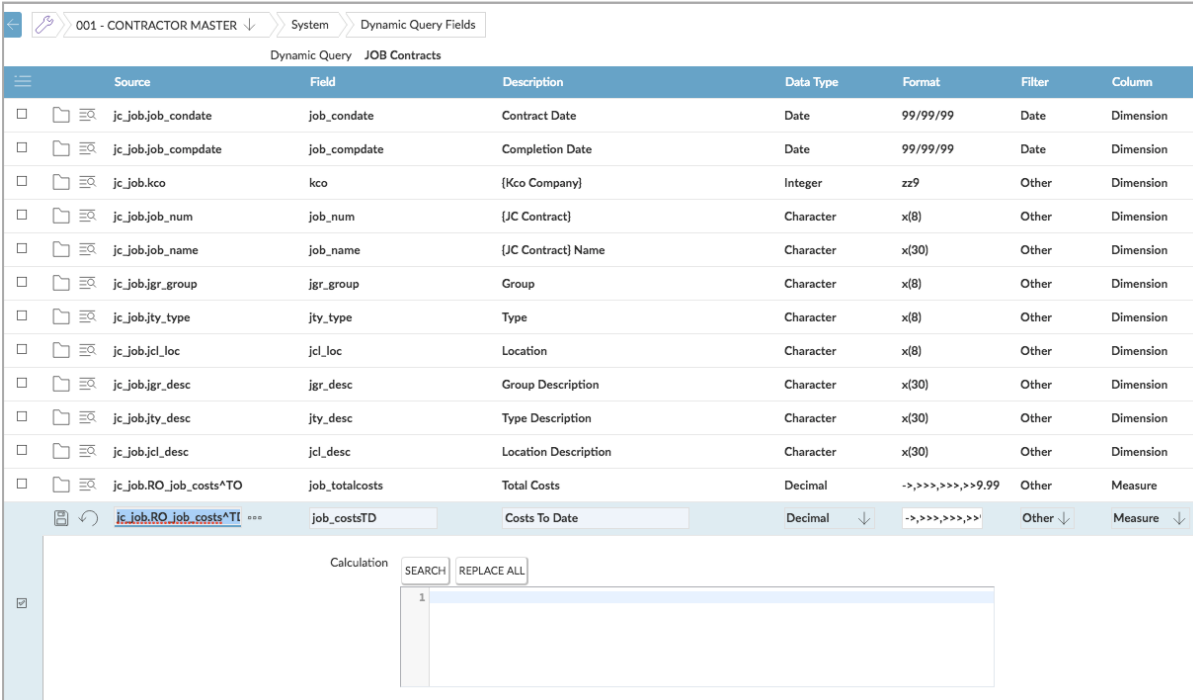
A list of the fields on the update form specified in **Parameters Section**. These will be prompted for in the Parameters section of the selection form.

## Dynamic Filters

The number of dynamic filters that will be built on the Filters section of the selection form for the enquiry or report.

When you have defined the table and query, follow the link on the **Query** code to specify the fields that will be available.

Figure 10: Dynamic Query Fields



Source	Field	Description	Data Type	Format	Filter	Column
jc_job.job_condate	job_condate	Contract Date	Date	99/99/99	Date	Dimension
jc_job.job_compdate	job_compdate	Completion Date	Date	99/99/99	Date	Dimension
jc_job.kco	kco	{Kco Company}	Integer	zz9	Other	Dimension
jc_job.job_num	job_num	{JC Contract}	Character	x(8)	Other	Dimension
jc_job.job_name	job_name	{JC Contract} Name	Character	x(30)	Other	Dimension
jc_job.jgr_group	jgr_group	Group	Character	x(8)	Other	Dimension
jc_job.jty_type	jty_type	Type	Character	x(8)	Other	Dimension
jc_job.jcl_loc	jcl_loc	Location	Character	x(8)	Other	Dimension
jc_job.jgr_desc	jgr_desc	Group Description	Character	x(30)	Other	Dimension
jc_job.jty_desc	jty_desc	Type Description	Character	x(30)	Other	Dimension
jc_job.jcl_desc	jcl_desc	Location Description	Character	x(30)	Other	Dimension
jc_job.RO_job_costs*TO	job_totalcosts	Total Costs	Decimal	->,>>>,>>>,>>9.99	Other	Measure

Calculation
1



## Fields

### Source

The source table.field for this field in the query. The table.field combination must exist in the corresponding RSP defined for this table on the query. Leave this blank to specify a calculation (see below).

### Field

The field name in the resulting temp table. It must be unique for this query. Although you can use any name for the field, it is sensible to use standard names (for example: job\_num), as the name dictates which lookup is used.

### Description

The description (label) for the field.

### Data Type

The data type of the field.

### Format

The default format for the field.

### Filter

Whether the field should be presented to the user in the **Date Filters** section or the **Filters** section. Leave this blank if you do not want the field to appear as a filter.

### Column

Whether the field should be presented to the user as group field (dimension) or a value field (measure). Blank will mean the field can be selected for a report/extract but cannot be used in the grouping report.

### Calculation

A calculation to populate the field in the temp table. This is run if the **Source** is blank. Calculations must use the original source fields from the query tables, not the calculated fields.

Do not use variables that attempt to span across fields as the user may not select all the fields you use. Variables can be used within the calculation of a single value.

## 6.4.2 Dynamic Enquiry


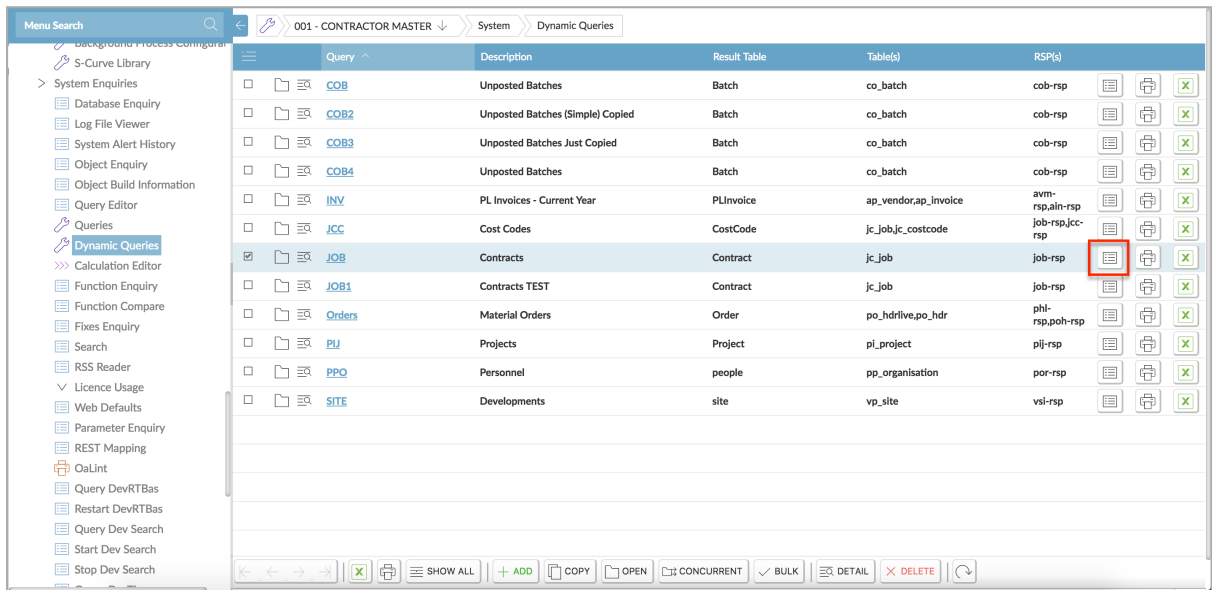
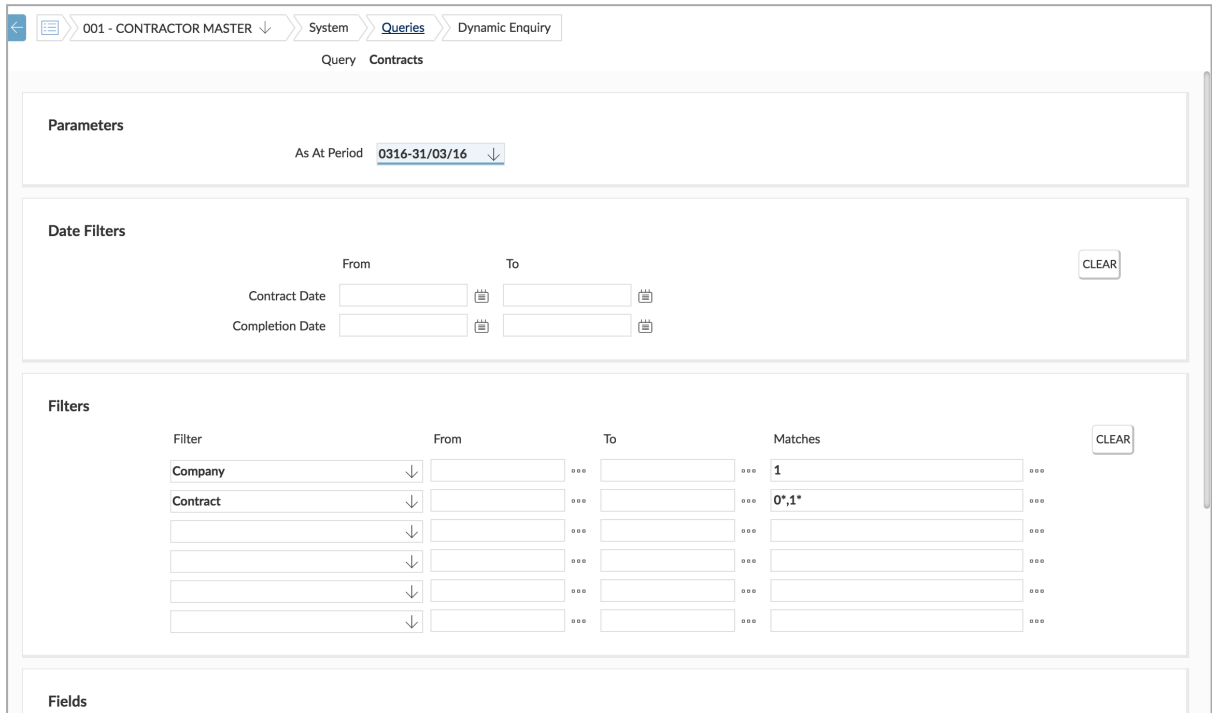
Dynamic enquiries can be run from the *Dynamic Query* browse by clicking the  button.

Figure 11: Dynamic Query Enquiry



The selection for the enquiry is split into four sections (though some sections may not be used).

**Figure 12: Dynamic Query Enquiry Selection Screen**



001 - CONTRACTOR MASTER > System > Queries > Dynamic Enquiry

Query Contracts

**Parameters**

As At Period: 0316-31/03/16

**Date Filters**

From To CLEAR

Contract Date

Completion Date

**Filters**

Filter	From	To	Matches
Company	<input type="text"/>	<input type="text"/>	1
Contract	<input type="text"/>	<input type="text"/>	0*,1*
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

CLEAR

**Fields**

**Parameters** are single values that are required to run the query or fields, for example: Financial Period.

**Date Filters** are fields that are specified as date filters in the dynamic query configuration and are shown in full, with a label, **From** and **To** fields, and a date lookup. The CLEAR button to the right will clear all the fields in this section.

The fields in the **Filters** section are dynamic: you can select the field and the filter value. The number of dynamic filters shown is defined in the dynamic query configuration. The lookup button is active if the field selected in the dynamic filter has a lookup associated with it. The CLEAR button to the right will clear all the fields in this section. The filters can include filters on ANY of the fields in the dynamic query even if they are RO fields or are calculations.

The **Fields** section allows you to select which fields from the dynamic query will be shown in the enquiry page.

Click the → Next button to run the enquiry and show the resulting records and fields.



Figure 13: Dynamic Query Enquiry Results

001 - CONTRACTOR MASTER						
System > Queries > Dynamic Enquiry						
	Company	Contract	Contract Name	Group	Group Description	Costs To Date
<input type="checkbox"/>	1	000004	antshy wbs only act37383939303	00	Sector 00X	0.00
<input type="checkbox"/>	1	000005	antshy (wbs - only sect)	00	Sector 00X	0.00
<input type="checkbox"/>	1	000006	antshy (wbs - sect+act)	LB2	Sector LB2	223,704.20
<input type="checkbox"/>	1	000007	antshy (non-wbs)	LB2	Sector LB2	1,739.00
<input type="checkbox"/>	1	000012	eu-NoWbs NOT change anything h	00	Sector 00X	11,057.98
<input type="checkbox"/>	1	000016	Plant tests, Full Allocation	00	Sector 00X	0.00
<input type="checkbox"/>	1	000017	Plant tests, GRN	00	Sector 00X	0.00
<input type="checkbox"/>	1	000018	Plant tests, Order	00	Sector 00X	0.00
<input type="checkbox"/>	1	00002	YL-Test02-Tender	01	Sector 01	0.00
<input type="checkbox"/>	1	09876	Skynet	00	Sector 00X	0.00
<input type="checkbox"/>	1	1	Aberdeen City Council, No WBS,	00	Sector 00X	81,862.01
<input type="checkbox"/>	1	12222222	antshy test	00	Sector 00X	10.00
<input type="checkbox"/>	1	123	Contract 123X	00	Sector 00X	62.00
<input type="checkbox"/>	1	1234	test	00	Sector 00X	32,476.28
<input type="checkbox"/>	1	12345678	123456	00	Sector 00X	0.00
<input type="checkbox"/>	1	1235	The Wharf (1235)	LB2	Sector LB2	66.00
<input type="checkbox"/>	1	1236	1236	00	Sector 00X	12,3456,789,250.00

Navigation: [Back] [Forward] [Refresh] [Print] [SHOW ALL] [REGENERATE] [BACK]

### 6.4.3 Report


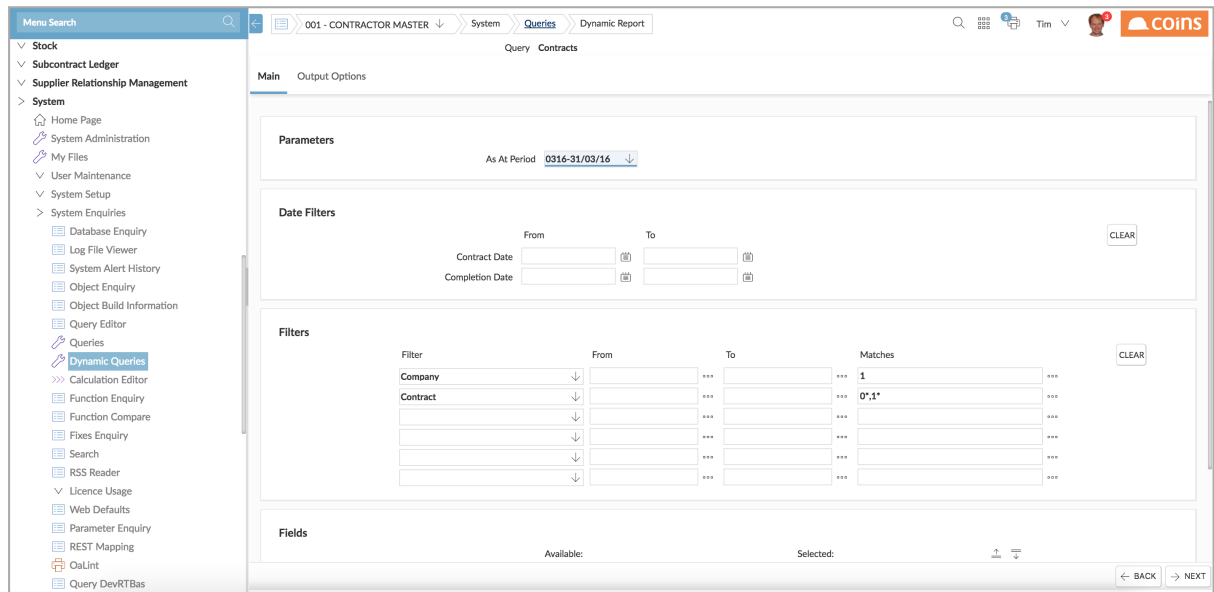
Dynamic reports can be run from the *Dynamic Query* browse by clicking the  button. The selection screen is the same as for the Dynamic Enquiry.

Figure 14: Dynamic Report Selection Screen



The report runs and (if requested) produces a PDF and spreadsheet export, and the pivot data for the resulting records and fields.



Figure 15: Dynamic Report Output

**System - Dynamic Report**  
**CONTRACTOR MASTER**

---

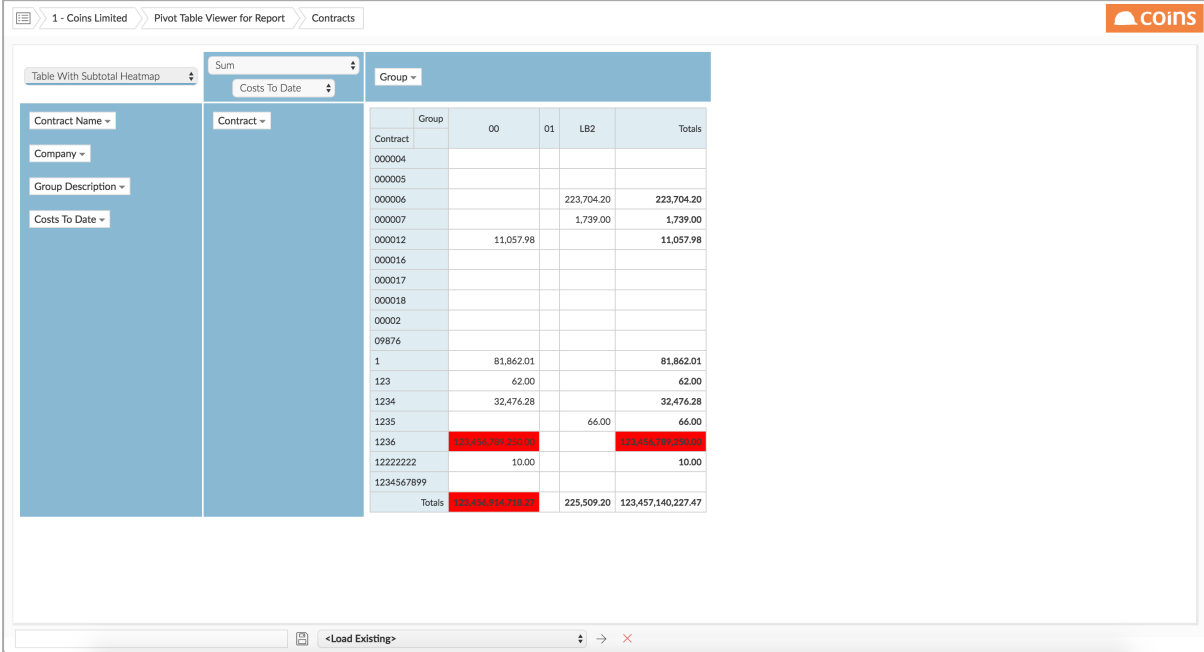
Con	Contract	Contract Name	Group	Group Description	Costs To Date
1	000004	antsy wbs only act3738393930303y3y3y	00	Sector 00X	0.00
1	000005	antsy (wbs - only sect)	00	Sector 00X	0.00
1	000006	antsy (wbs - sect+act)	LB2	Sector LB2	223,704.20
1	000007	antsy (non-wbs)	LB2	Sector LB2	1,739.00
1	000012	eu-NoWbs NOT change anything here	00	Sector 00X	11,057.98
1	000016	Plant tests, Full Allocation	00	Sector 00X	0.00
1	000017	Plant tests, GRN	00	Sector 00X	0.00
1	000018	Plant tests, Order	00	Sector 00X	0.00
1	00002	YL-Test02-Tender	01	Sector 01	0.00
1	09876	Skynet	00	Sector 00X	0.00
1	1	Aberdeen City Council, No WBS, GRN/ FullAlloc	00	Sector 00X	81,862.01
1	12222222	antsy test	00	Sector 00X	10.00
1	123	Contract 123X	00	Sector 00X	62.00
1	1234	test	00	Sector 00X	32,476.28
1	1234567899	123456	00	Sector 00X	0.00
1	1235	The Wharf (1235)	LB2	Sector LB2	66.00
1	1236	1236	00	Sector 00X	12,3456,789,250.00



Figure 16: Dynamic Report Spreadsheet

	A	B	C	D	E	F	G	H
1	Report:	%WSY1353RXXX-System - Dynamic Report						
2	Date/Time:	16/10/16	08:26					
3	Selection:	&companySelector=1&RS_glp_fdate__1=31%2F03%2F16&RS_Dynamic=Y&F						
4	Company:	1						
5	User:	TIMARM						
6								
7	RecordType	kco	job_num	job_name	jgr_group	jgr_desc	job_costsTD	
8	ttContract	1	000004	antsy wbs c	00	Sector 00X	0	
9	ttContract	1	000005	antsy (wbs	00	Sector 00X	0	
10	ttContract	1	000006	antsy (wbs	LB2	Sector LB2	223704.2	
11	ttContract	1	000007	antsy (non-	LB2	Sector LB2	1739	
12	ttContract	1	000012	eu-NoWbs N	00	Sector 00X	11057.98	
13	ttContract	1	000016	Plant tests, F	00	Sector 00X	0	
14	ttContract	1	000017	Plant tests, C	00	Sector 00X	0	
15	ttContract	1	000018	Plant tests, C	00	Sector 00X	0	
16	ttContract	1	00002	YL-Test02-Te	01	Sector 01	0	
17	ttContract	1	09876	Skynet	00	Sector 00X	0	
18	ttContract	1	1	Aberdeen Cit	00	Sector 00X	81862.01	
19	ttContract	1	12222222	antsy test	00	Sector 00X	10	
20	ttContract	1	123	Contract 123	00	Sector 00X	62	
21	ttContract	1	1234	test	00	Sector 00X	32476.28	
22	ttContract	1	1234567899	123456	00	Sector 00X	0	
23	ttContract	1	1235	The Wharf (1	LB2	Sector LB2	66	
24	ttContract	1	1236	1236	00	Sector 00X	1.23457E+11	
25								

Figure 17: Dynamic Report Pivot Table



Contract	Group	00	01	LB2	Totals
000004					
000005					
000006				223,704.20	223,704.20
000007				1,739.00	1,739.00
000012		11,057.98			11,057.98
000016					
000017					
000018					
00002					
09876					
1		81,862.01			81,862.01
123		62.00			62.00
1234		32,476.28			32,476.28
1235				66.00	66.00
1236		22,457,227.47			22,457,227.47
12222222		10.00			10.00
1234567899					
<b>Totals</b>		<b>225,509.20</b>	<b>225,509.20</b>	<b>123,457,140,227.47</b>	



### 6.4.4 Dynamic Grouping

Dynamic grouping is similar to the dynamic report, except that you are able to select fields from the Dimension fields and values from the Measure fields as defined in the configuration. The resulting output is aggregated by the dimensions, and shows the count of the records for that grouping and the sum of the value fields selected.


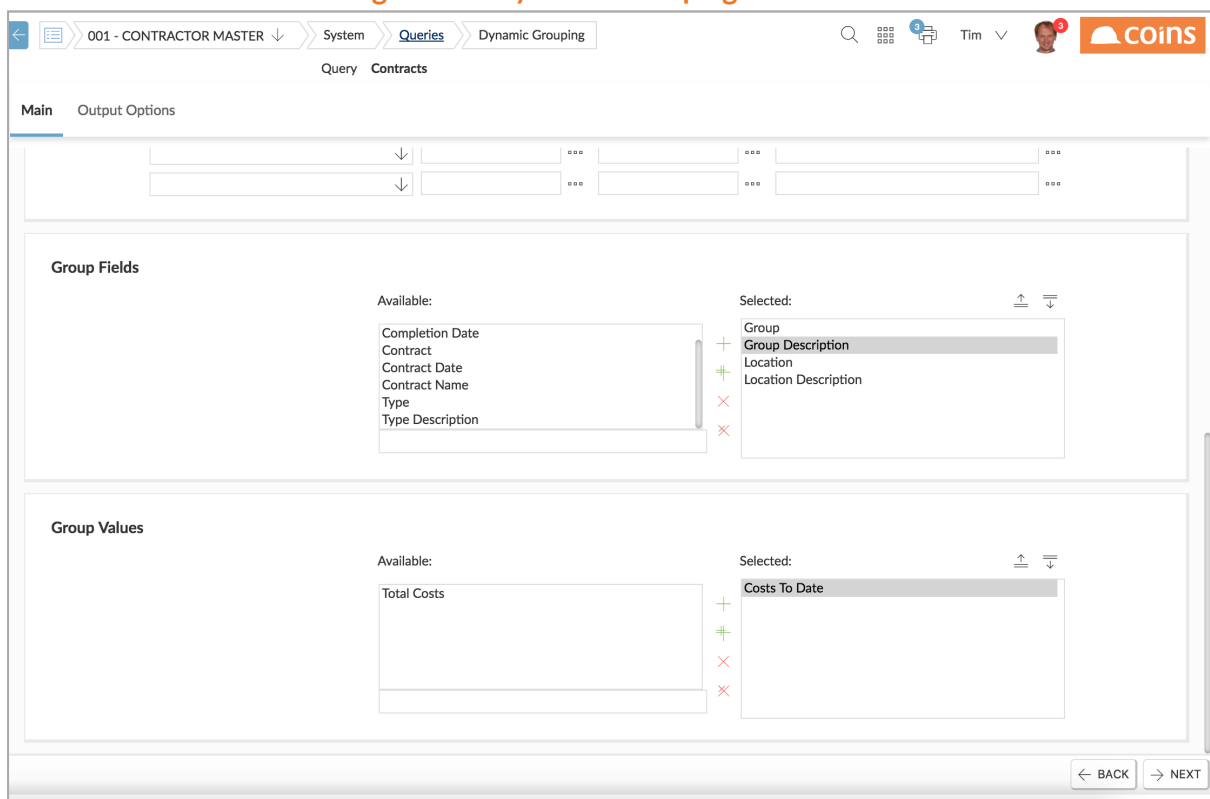
Dynamic grouping can be run from the *Dynamic Query* browse by clicking the  button.

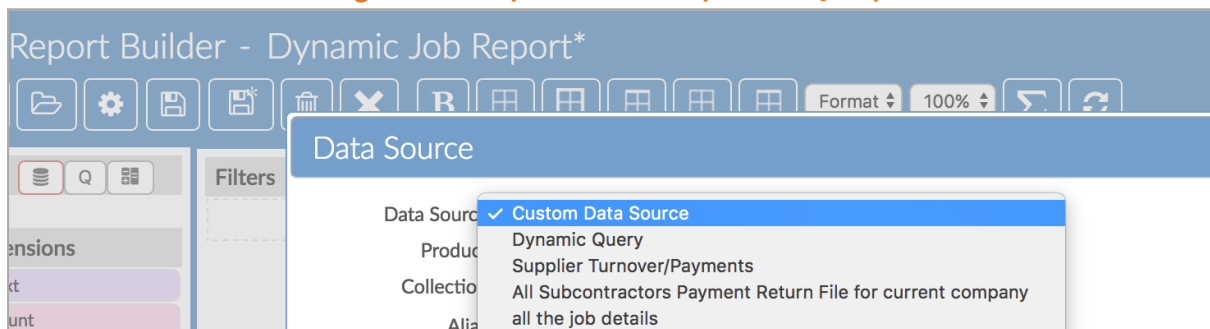
Figure 18: Dynamic Grouping Selection



## 6.4.5 Report Builder

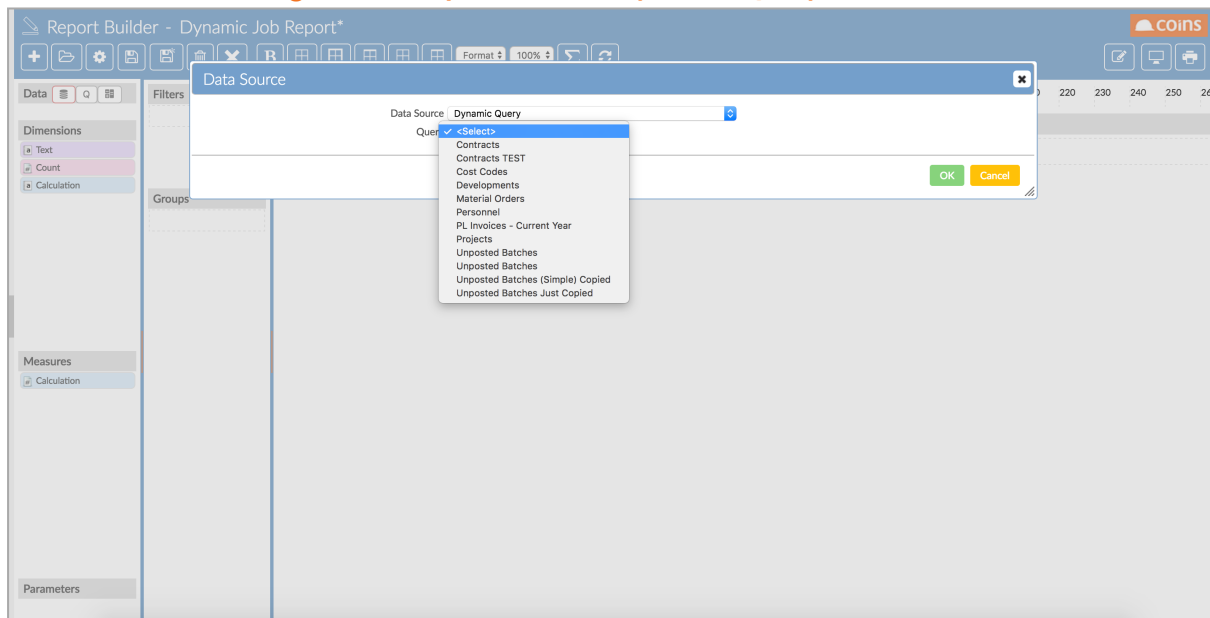
Dynamic queries have been added as a data source to **Report Builder**. The option appears in the list of data sources underneath custom data source (if a compile licence is present) and before all the standard data sources that have been defined using datasets and data source maintenance.

Figure 19: Report Builder Dynamic Query



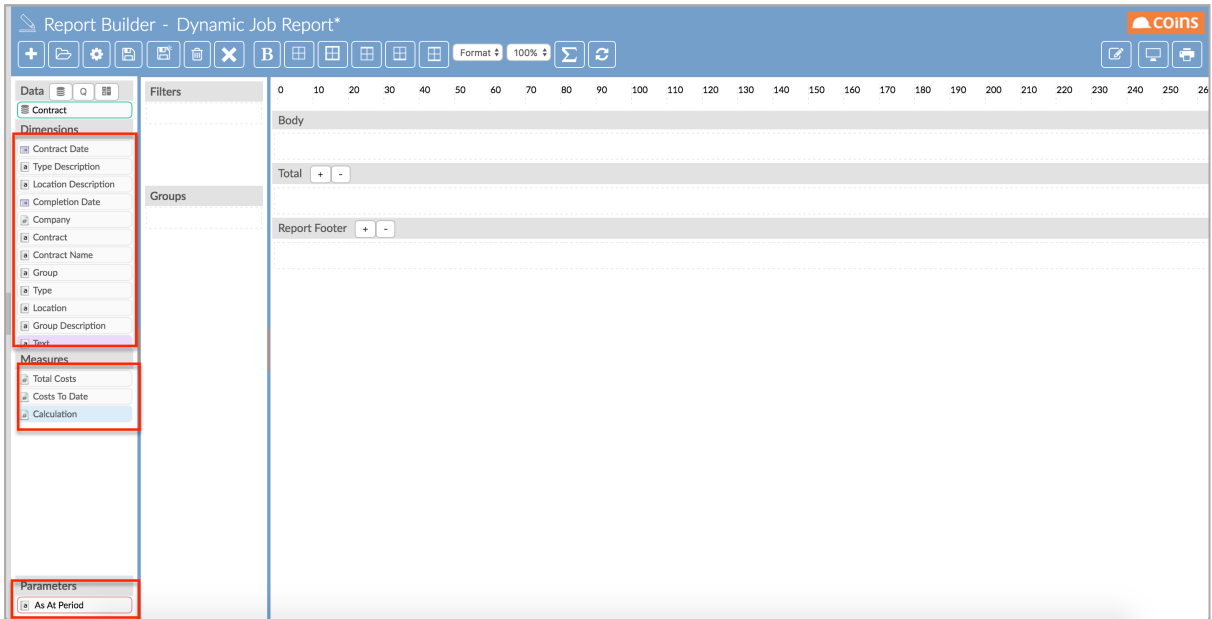
After you select Dynamic Query, you are presented with a list of dynamic queries to which you are allowed access (see 1, Dynamic Queries - Query Security).

Figure 20: Report Builder Dynamic Query Selection



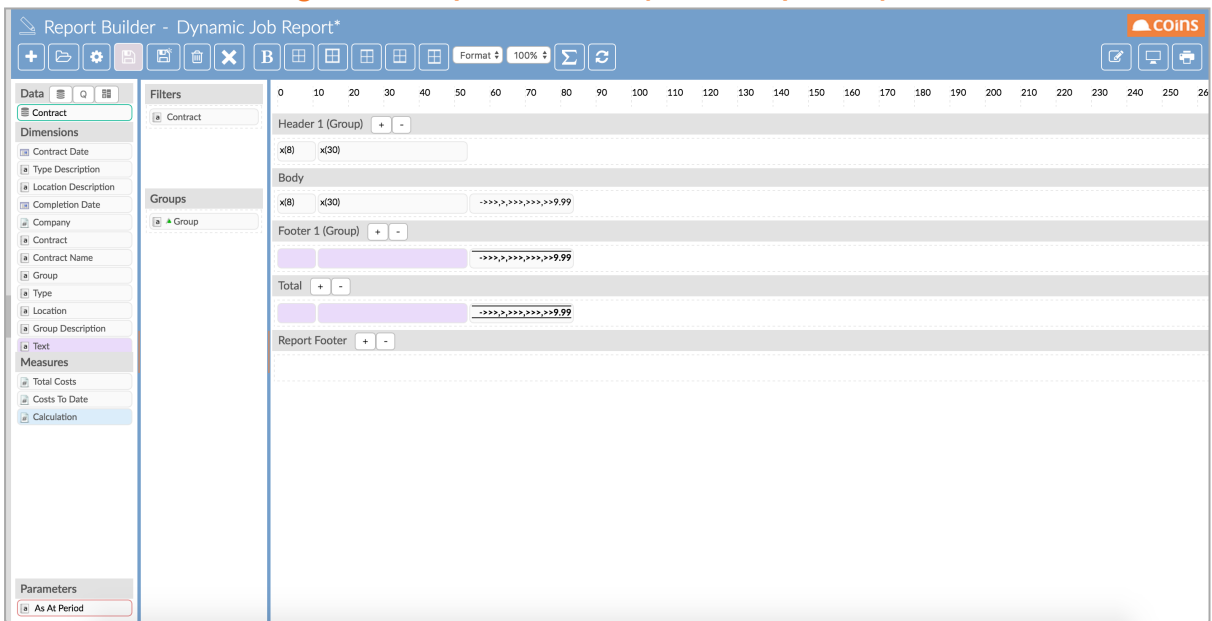
The resulting dataset from the dynamic query is shown in the selection panel of the report builder. Dimension fields and Measure fields are shown separately. Any parameters configured on the dynamic report need to be set in the Parameters section at the bottom.

Figure 21: Report Builder Dynamic Query Fields



You can then drag fields to the canvas, filters and groups.

Figure 22: Report Builder Dynamic Report Layout



The normal report enquiry and output is available.



Figure 23: Report Builder Dynamic Report Output

Report Builder - Dynamic Job Report\*

Format: 100%

Dynamic Job Report  
CONTRACTOR MASTER

Contract/Contract Name	Costs To Date
sk EK Simio	0.00
eleuspG eleusp GRN+WBS	5,777.50
eleuspO eleusp Order + WBS	48,543.48
eleuspO eleusp Order + WBS	60.00
	<b>12,345,063,764.68</b>
01 Sector 01	
00002 VL_Treat12-Tender	0.00
hal-005 hal-005 name	0.00
hal-002 hal-002	200,105.85
hal-001 hal-001	56,480.78
hal-003 hal-003	382.89
hal-006 hal-006 name	0.00
hal-007 hal-007 name	0.00
	<b>255,969.52</b>
LB2 Sector LB2	
1235 The Wharf (1235)	66.00
000007 antshy (non-wbs)	1,742.00
000006 antshy (wbs - sect+act)	227,866.20
	<b>229,674.20</b>
	<b>12,345,549,208.40</b>

Parameters  
As At Period

## 7 OA Designer - Overview

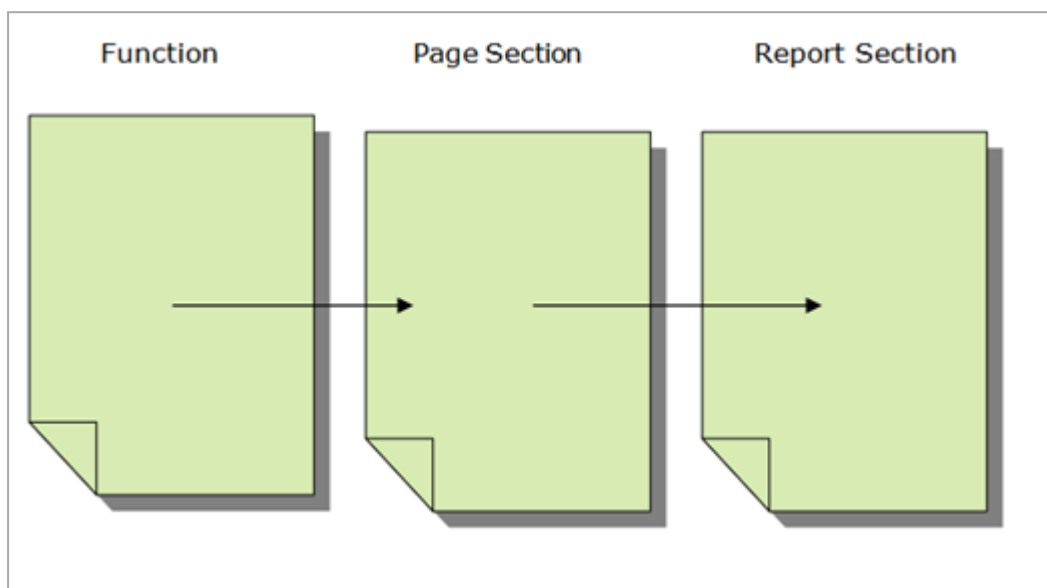
The Designer is a highly flexible reporting and screen design tool. It is aimed at more technically adept users who wish to develop more sophisticated reports, screen queries and customised user interfaces against the COINS database. Free format queries can be created and a comprehensive data dictionary and catalogue is provided to enable you to create these queries and identify appropriate data fields. You can also write personal data exports.

### 7.1 Designer key features

- Copy and amend standard reports
- Build simple and complex reports
- Use simple and complex calculations
- Use conditional formatting
- Use graphs, graphical displays, KPI reporting and trending
- Display/update user defined data including report stores
- Create automated report packs and distribution
- Create Data Marts
- Integrate with all generic reporting features
- Integrate with Workflow

## 7.2 Functions and Sections

To generate a Page or Report in coins OA there are three development components :



A function in OA simply runs a procedure (unless defined as a menu).

This procedure is nearly always wou005.p and it simply builds a webpage for display in coins OA.

To determine what is displayed and how it is displayed, coins OA (wou005.p) uses Pages.

A Page Design contains Form definitions for the webpage. These Forms definitions may include headers, bodies, footers, totals etc. Each of these Form Types will also have Fields defined for display.

In addition, Pages also allow for the definition of Filters which will be available on the webpage when the function is run.

When you are running a Report in coins OA, the Function will still reference a Page, however this webpage will contain only the information (layout and fields required) for the report selection criteria. The actual definition of fields, layout, headers and totals are then maintained in the Report Design.

When you create a new function in coins OA – until you have completed the Page Design nothing will be returned to screen. If this function is to be a report, it is also necessary to define the Report Design.

## 7.3 Function Naming

Below is the current COINS standard naming conventions for OA functions

example: %WPL5100BAVMT1

%WXXnnnmCTTT[Y]Tn

%	% Denotes a coins standard function (which will be updated and maintained by coins and will be overwritten as standard data during processes such as environment upgrades). If you wish to amend a standard Coins function, it is strongly suggested, as with Coins+ you copy the function and rename as with the original but replacing the % with a +.
XX	Module code (i.e. GL, JC or PO)
nnn	functions sequence number (Enquiries generally begin 5nn and Reports 3nn)
m	sub-sequence number (often 0 but used for imbedded/inline reports)
C	Class of page (Browse, Form, List, Report, Summary, Utility, Web Page)
TTT	Table ID (From the main table used – see Database Enquiry)
Y	(optional) Sub-function type (for buttons etc.) (Add, Update, Delete, More, eXport, Generate, Bulk, Options Menu, Link/Actions Menu), Concurrent Update
T	Tab
N	Sequence

## 7.4 Function Parameters

Function menu parameters are used extensively in COINS OA, the function menu parameter can pass information as part of the request to COINS, for example a financial date to a sub-query in a dig deeper, or to define where the request should look for information, for example a specific report section.

When adding parameters to a function you add field ={value}.

There must be NO spaces in the statement and any additional functions must be prefixed with an '&' symbol. The statement should NOT contain any special characters including quotes, semicolon etc.

Example:

```
stn_code=ABCXXX&EndPeriod={selectEndPeriod}
```

The function menu parameters will be appended to the URL passed through the COINS Web Agent.



### 7.4.1 stn\_code

You can point a function to a different Page by entering **stn\_code=page-code** in the Parameters field of the function.

Example

```
stn_code=%WSYBSUR1
```

This allows you to set up standard page layouts and share them between some different but similar functions.

## 7.4.2 rtn\_code

You can point a function to a different report section by entering

```
rtn_code=report-section-code
```

in the Parameters field of the function. This allows you to set up standard report layouts and share them between some different but similar reports.

It is possible to create a function and a page containing report selection criteria that can run multiple reports from one menu option. This may be useful when scheduling multiple reports with standard selection –

To do this simply enter a Function parameter as

```
rtn_code={comma separated list of reports}
```

### 7.4.3 Exptype

ExpType= XML - XML Output

CSV Output if space or any other text

Examples:

ExpType=XML

ExpType=CSV

This parameter overrides the system parameter 'EXPTYPE' and forces either XML or CSV output for the Excel file. This may be useful if, for example, the system is configured for CSV but you require XML output to preserve leading zeros and to preserve number strings as character output.

Whilst the parameter will allow a space to be used to denote CSV output, it is good practice to specify the text CSV - even though this is not actually validated. This will clearly indicate the format being used and, if this is the last parameter in a list, will prevent another user adding a new parameter to the end of the ExpType= and invalidating the test.

## 7.4.4 noInfo

noInfo=Y

Suppresses the 1st 6 Rows and 1st Columns and Sort Columns from the Excel output

Function	Description	Type	Module	Category
LKTEST1	LK Test Noinfo	Function	JC-JC Contract Status	REP-Report

Context: LK Test Noinfo  
Program: wou005  
Parameters: noinfo=Y  
Role Type: -

Figure 24: Result with Parameter

	A	B	C	D	E	F	G
1	Contract	Name	Costs To Date	Revenue	Profit	Margin	
2	777888	ANTFOM F-1 Circle	15	0	-15	0	
3	C4123	Laing O'Rourke	0	0	0	0	
4	ng1	ng1	3695.2	0	-3695.2	0	
5	ng2	ng2	-15	0	15	0	
6	v7n	Harefield Lakeside	9382.46	0	-9382.46	0	
7	v7w	Dublin Docks	496253.48	0	-496253	0	
8							

Figure 25: Result without Parameter

	A	B	C	D	E	F	G	H	I	J	K	L
1	Report:	LKTEST2-JC Contract Status - LK Test Noinfo not set										
2	Date/Time:	15/10/10 11:14										
3	Selection:	RS_job_num_1=&RS_job_num_2=&RS_job_num_3=&RS_jgr_group_1=&RS_jgr_group_2=&										
4	Company:	1										
5	User:	LESOLI										
6												
7	RecordType	Group	Typet	Loct	Contract	Contract	Name	Costs To	Revenue	Profit	Margin	
8	jc_job	00	CCC	ABERC	777888	777888	ANTFOM F-1 Circle	15	0	-15	0	
9	jc_job	00	FM	ESSEX	C4123	C4123	Laing O'Rourke	0	0	0	0	
10	jc_job	00	WIP	BERKS	ng1	ng1	ng1	3695.2	0	-3695.2	0	
11	jc_job	00	WIP	BERKS	ng2	ng2	ng2	-15	0	15	0	
12	jc_job	00	WIP	BERKS	v7n	v7n	Harefield Lakeside	9382.46	0	-9382.46	0	
13	jc_job	00	WIP	BERKS	v7w	v7w	Dublin Docks	496253.5	0	-496253	0	

## 7.4.5 ExportType

ExportType=

- X - XML Output
- S - Spreadsheet
- P - PDF

This parameter may be used to restrict the output types for reports. By default COINS OA will generate all three report types. Note this does not change the output format of the file, only which of the output types are created.

Examples:

**ExportType=S** (only the Excel file is generated, no PDF or XML)

**ExportType=P,S** (Both PDF and Excel are generated, no XML)



## 7.4.6 Sidemenuhelp

This suppresses the Menu Frame and the Help Frame from the Screen to allow the best use of the data section of the OA Screen

sidemenuhelp=0,\*0

## 7.4.7 Passing Variables

You can default variables values into fields by adding them to the Parameter of a Function.

Example:

**RS\_Date={co\_config.RO\_dateOffset^0}**

will default today's date into a field.

Please note that this will only work once the function has been added to a menu, it will not work if the function is run from the URL via the mainarea functionality



## 7.5 URL Parameters

Delete this text and replace it with your own content.



## 7.5.1 Helpmode

HelpMode=

- **layout** - To show the cell borders, add &helpmode=layout to the url of the page, and refresh it.
- **stdpage** - Adding &helpmode=stdpage to the url will force framework to use the % page. It is a useful feature when client uses a + page.
- **stack** - If &helpmode in the url contains "stack" then the procedure stack will be included with error messages.
- **prompt** - Will display field names, function codes etc in the Side Frame Help.
- **clearDefault** - Adding &helpmode=clearDefault to the url will clear any held filters held for a page. Particularly useful if the stored data is inconsistent with the current page definition and it has resulted in either a blank page , errors, or missing bits of HTML. This helpmode option will remove all filters and sort orders and will reset the page as though it were the first time in the system. This will remain in the URL in each page until you remove it. (Avail from patch 0901)
- **edit** - This will enable you to change the side-bar help to enable you to add comments against fields
- **COINSInfo** - To see debug information on web page, add &COINSInfo=true to the url of the page. This has the same effect as <CTRL>+<SHIFT>+C

## 7.6 Forms Service Procedures and Report Selection Generates

Form Service Procedures are additional procedures used in generating coins OA Reports in each module

They also provide standard report selection criteria which can then be used in Report Queries. This section lists the commonly used FSP's and the Selection Generates available.

## 7.6.1 Bill of Quantities - BQFREP

Program : bqfrep.p

Description	Tables	Page Generate	Query Replace
Plot	vp_wbsdef	iwPlotSelectionGenerate	{iwPlotSelect}
CostHead	jc_stndcc, vp_wbsbud	jscSelectionGenerate	{jscSelect}

## 7.6.2 Cash Book - CBFREP

Program : cbfrep.p

Description	Tables	Page Generate	Query Replace
Sector	co_vattrans,	ganSelectionGenerate	{ganSelect}
Source	co_vat,cb_tdet,co_batch, gl_trans, jc_inctrans, jc_costtran	cojSelectionGenerate	{cojSelect}
Sales Account	ar_cust	rcmSelectionGenerate	{rcmSelect}
VAT	co_vat	vatSelectionGenerate	{vatSelect}

### 7.6.3 Central Repository - CIFREP

Program : cifrep.p

Description	Tables	Page Generate	Query Replace
Company	ci_company	cimSelectionGenerate	{cimSelect}
Office	ci_office	cioSelectionGenerate	{cioSelect}

## 7.6.4 Company - COFREP

Program : cofrep.p

Description	Tables	Page Generate	Query Replace
Source	co_vattrans, cb_tdet, co_batch, gl_trans, jc_inctrans, jc_costtran	cojSelectionGenerate	{cojSelect}

## 7.6.5 Credit Control - CNNFREP

Program : cnnfrep.p

Description	Tables	Page Generate	Query Replace
Sector	ap_invoice	ganSelectionGenerate	{ganSelect}
	ap_invopen		{ganSelectInvOpen}
	ap_chqopen		{ganSelectChqOpen}
	sc_chqopen		{ganSelectChq}
	sc_certopen		{ganSelectCertOpen}
Contract	jc_job	jobSelectionGenerate	{jobSelect}
Phase	jc_costcode, jc_phase	jphSelectionGenerate	{jphSelect}
CostCode		jphSelectionGenerate	{jccSelect}

## 7.6.6 Contract Sales - CSFREP

Program : csfrep.p

Description	Tables	Page Generate	Query Replace
Sales Account	ar_cust	rcmSelectionGenerate	{rcmSelect}
Contract	jc_job	jobSelectionGenerate	{jobSelect}
Phase	jc_costcode, jc_phase	jphSelectionGenerate	{jphSelect}
Sector		ganSelectionGenerate	{ganSelect}
Used with generate programs for Sort Order		jobSortSelectionGenerate	
CostCode	jc_costcode	jccSelectionGenerate	{jccSelect}



## 7.6.7 Fixed Assets - FAFREP

Program : fafrep.p

Description	Tables	Page Generate	Query Replace
Fixed Asset	fa_asset	faaSelectionGenerate	{faaSelect}

## 7.6.8 Facilities Management - FMFREP

Program : fmfrep.p

Description	Tables	Page Generate	Query Replace
Small Works Job	sw_job	swjSelectionGenerate	{swjSelect}
Contract	jc_job	jobSelectionGenerate	{jobSelect}
Phase	jc_costcode, jc_phase	jphSelectionGenerate	{jphSelect}
Small Works Invoice	sw_invhead	sihSelectionGenerate	{sihSelect}
Schedule of Rates	sw_sor	sowSelectionGenerate	{sowSelect}
CostCode	jc_costcode	jccSelectionGenerate	{jccSelect}



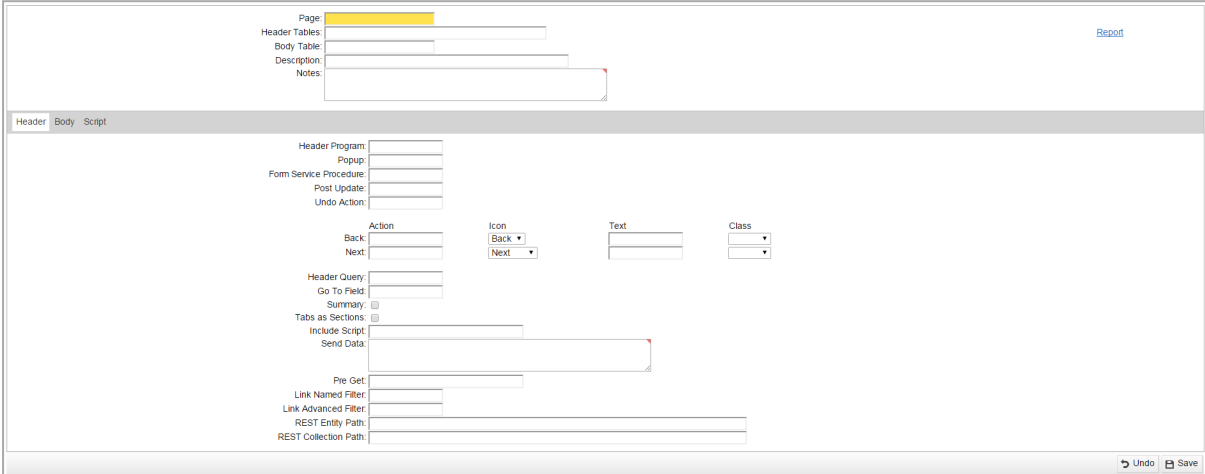
## 7.6.9 General Ledger - GLFREP

Program : glfrep.p

Description	Tables	Page Generate	Query Replace
GL Account	gl_acct	glaSelectionGenerate	{glaSelect}
Source	co_vat, cb_tdet, co_batch, gl_trans, jc_inctrans, jc_costtran	cojSelectionGenerate	{cojSelect}
Currency		curSelectionGenerate	
Batch	co_batch	cobSelectionGenerate	{cobSelect}
Destination Batch	co_batch	cobLinkedSelectionGenerate	{cobLinkedSelect}
Batch Status	co_batch	cobStatusSelectionGenerate	{cobStatusSelect}

## 7.7 Page Designer

To create a new Page Design click on the Add button . The following page will be dispalyed.



## 7.7.1 Page Header



Field	Description
Page	The name for this page. To associate a Page with a Function name the Page with the same code as the Function.
Header Tables	Header tables make fields and parameters available to the page; for example, co_config will make the currency fields available, gl_config will make periods available.
Body Tables	Body Table is a database table that the body of this page displays. Only the Body Table can be updated.
Notes	These notes are for documenting advice/guidance regarding the purpose of the page. They play no part in the building of the page.

The Header and Body Tables assigned to a page will dictate which RSP's are accessed by the query, and as such will provide certain business logic to the Page. It is recommended that co\_config (COINS Company configuration) and the configuration table of the Module being queried, for example jc\_config are used as Header Tables.

## 7.7.2 Page Header Tab

This tab shows configuration for the header part of the page.

Field	Description
Header Program	The name of a Progress program to produce HTML to replace the header – used for special situations when the screen cannot be built using the standard toolkit.
Popup	An optional menu function that defines the entries on a combo in the page header. It is used to navigate to other screens.
Form Service Procedure	If this form passes data on to another procedure (such as a report), this is the name of a Form Service Procedure program that processes the data
Post Update	<p>The code for the function to go to after the add or update is complete.</p> <p>For example, after adding a new user, you may want to go to the user summary to update the user's access.</p> <p>This can contain a number of questions each with a function and a final default function. If the default function is missing then the browse is redisplayed.</p> <p>For example, %WVPBVSC/SC701,%WVPBXXX/SC702,%WSCSSBS would ask two questions SC701 and SC702; if the answer is Yes then it would run the appropriate function. If the answer to both is No then it would run %WSCSSBS.</p> <p>You can also run the functions without display (in the post frame). For example, you change a company status on a company record and want to ask the user if they wish to copy to all the offices. To do this pass back a third parameter (separated by "/"). For example %WCIU999/CI170/POST would run function %WCIU999 if CI170 was answered with Yes, but the output (there should be none) would be placed in the post frame and would not be visible. The next action in the POSTUPD series would then be run (if there was one) otherwise the list would be refreshed.</p>
Undo Action	A function to be run if the undo button is pressed

Field	Description
Back Action	<p><b>This overrides the standard undo behaviour</b></p> <p>If specified a "back" button is shown in the button bar. This is the function that will be run if that button is pressed.</p>
Back Icon	The icon to use for the back function
Back Text	The text to be used for the back function
Back Class	The class of the back button
Next Action	<p>The function code of the next function to display after this one. COINS displays the button to move on to this function.</p> <p>This can contain a number of questions each with a function and a final default function. If the default function is missing then the user will stay on the current page.</p> <p>For example, %WVPBVSC/SC701,%WVPBXXX/SC702,%WSCSSBS would ask two questions SC701 and SC702; if the answer is Yes then it would run the appropriate function. If the answer to both is No then it would run %WSCSSBS.</p> <p>You can also run the functions without display (in the post frame). For example, you might want to process a set of records after entering them and delete all which have zero value. To do this pass back a third parameter "/POST". You must not rely on this function being run as the user might choose to close the browser rather than pressing the next page button.</p>
Next Icon	The icon to use for the back function
Next Text	The text to be used for the back function
Next Class	The class of the back button
Header Query	<p>The function code used to point to the page section that contains the query that controls the navigation of the header tables on this page.</p> <p>If you enter a page section code, this enables navigation buttons on the summary page that allow the user to select further records in the query without having to go back to the browse screen.</p>
Go To Field	Whether on a page with header navigation a Go To field should be presented. This is the field to use to work out which record to Go To.
Summary	Whether the page is a summary page.

Field	Description
Tabs as Sections	<p>button. If this is not ticked, the page will open as an update page.</p> <p>Show tabs on fields as expandable sections</p>
Include Script	<p>If selected and tabs are specified on the fields in a form then they will be rendered as vertically expandable sections on the page.</p> <p>A comma-separated list of JavaScript files that you want to make available to the page</p>
Send Data	<p>A list of program and function (for example, vat-rsp.sendVATData) that populates a data table in JavaScript. This is for client-side scripting.</p>
Pre Get	<p>A method to call just before the web page is created. Specify the RSP name and the method to call (for example: gjh-rsp.buildTokens).</p> <p>This is called after all the header tables have been found in their respective RSPs and before the page HTML is created. It allows you to pre-process records before the page starts. For example, it is used to work out which tokens have been used in a GL standing journal - gjh-rsp.p / buildTokens</p>
Link Named Filter	<p>The code for a page that specifies a standard named filter to use on this page.</p>
Link Advanced Filter	<p>The code for a page that specifies a standard advanced filter to use on this page.</p>
REST Entity Path	<p>Path for REST mapping for an entity (single record)</p>
REST Collection Path	<p>Path for REST mapping for a collection (multiple records/query)</p>



### 7.7.3 Page Body Tab

This tab shows configuration for the body part of the page.

Field	Description
Body Program	The name of a Progress program to build the body section of the page
Body Rows	The number of rows of data to display on the page
Body Query	The query that retrieves the records to display on the page
Query Condition	A function that determines whether a record should be included or not. The function returns a logical value: yes to include the record, no to exclude it.
Ignore Security	This allows you to override the row security settings (for example, contract security, personnel security). Only root users can change this field.  For example, you could use this to provide a lookup on all employees, to update an "Employee's Manager" field, even if user security prevents users from accessing managers' records.
Sort Limit	The limit of the number of records that will be allowed before the sort options on the page will be suppressed. If you leave this as zero then sort options will always be available.
Body Title	The text for an optional subtitle that appears above a browse and below a header detail frame
Body Sum	A list of fields and variables to accumulate for the current body query.  This list is alternating pairs of a field (from the query tables) and a variable in which to store the total. Integer and Decimal values are totalled. Date, Logical and Character fields are shown if they are the same on each line. If you specify COUNT then a count is produced.
Actions	The function that will be used to build the options in Action selector for the browse list
Frozen Columns	The number of frozen columns in the browse
Ajax Update	Whether data updates (via AJAX) should be used in preference to screen refreshes
Matrix Columns	The number of "Grid Element" columns to display (if this is a matrix page)
Fixed Columns	The number of fixed columns to the left of a browse.  If non zero then horizontal and vertical scrolling of the table contents and labels is enabled. All records from the browse are shown.
Auto Add	If ticked, go into add mode if the browse is empty.
Condition	Option condition that must return true for the auto add to be active
Single Add	If ticked, go into add mode if the browse is empty, but stop after a single entry.



Field	Description
Sequenced Update	Whether updates are done sequentially; automatically opening the following record on a save
Graph Formatting	Extra formatting applied to the graph, if the browse is viewed as a graph.
Mandatory Columns	Columns that are mandatory in column sets
Essential Fields	<p>If you allow column sets on the page then this is the list of fields that will be made available if not included in the column set for an update.</p> <p>This allows you to specify fields that must be present during update so that e.g. lookups and validation continue work even if the field is not visible to user.</p>

Once this is complete Click Save . Once the Page Summary has been saved COINS presents additional Tabs.

Move to the Forms Tab once it becomes available.

## 7.7.4 Page Script Tab

Script shows the scripts for this page.

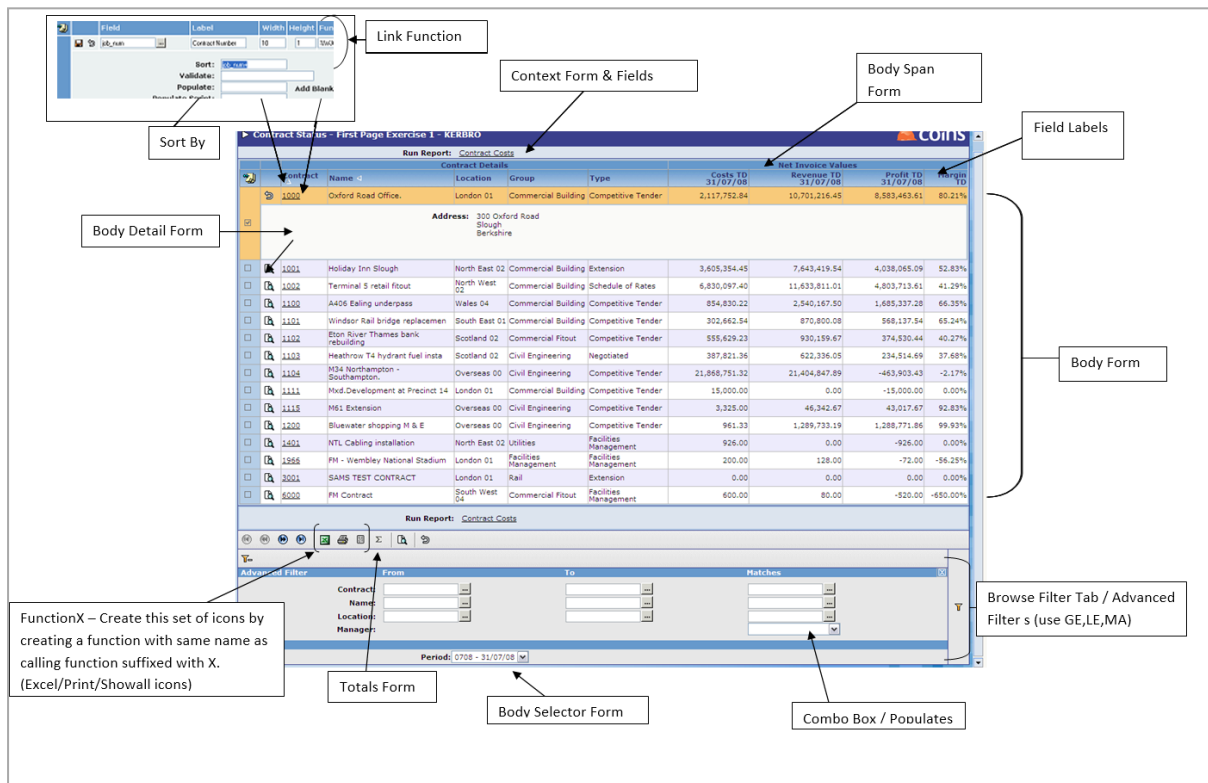
Field	Description
Page Script	The name of a script that will be run when the page is updated.
Update Script	This script is issued after a successful add, update or delete and is used to refresh a dependent browse on a separate tab.
Initial Calculation	Initial calculation that should be performed before the page is built.

## 7.7.5 Page Forms Tab

This Tab allows you to define which forms appear on the page.

Each page in coins OA has various elements, such as Titles, Footers and Body. In coins OA these are referred to as Forms. To display a simple listing enquiry, a Page must have at least a Body Form.

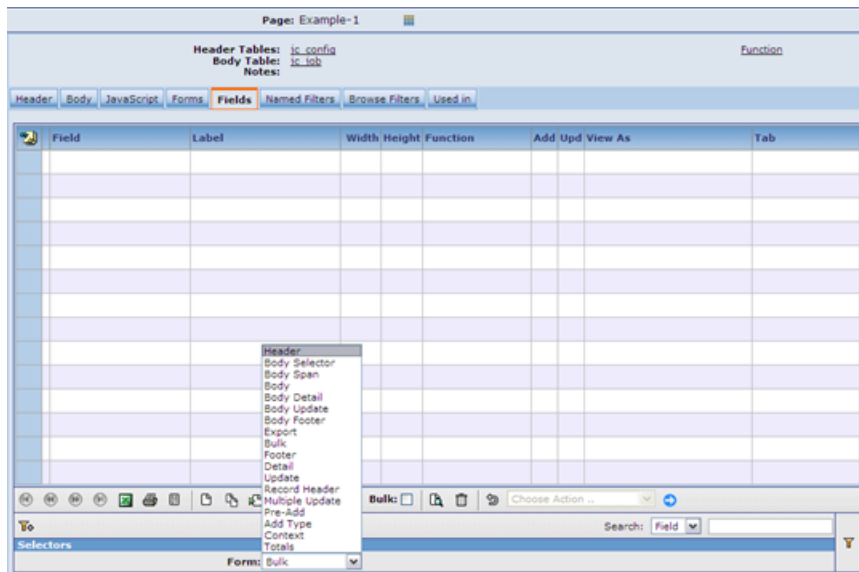
Different page forms display different types of information. For example, a context form displays fields from the parent record; an update form provides a form that allows you to update the record.



The screenshot shows the COINS BI Toolset interface with several annotated components:

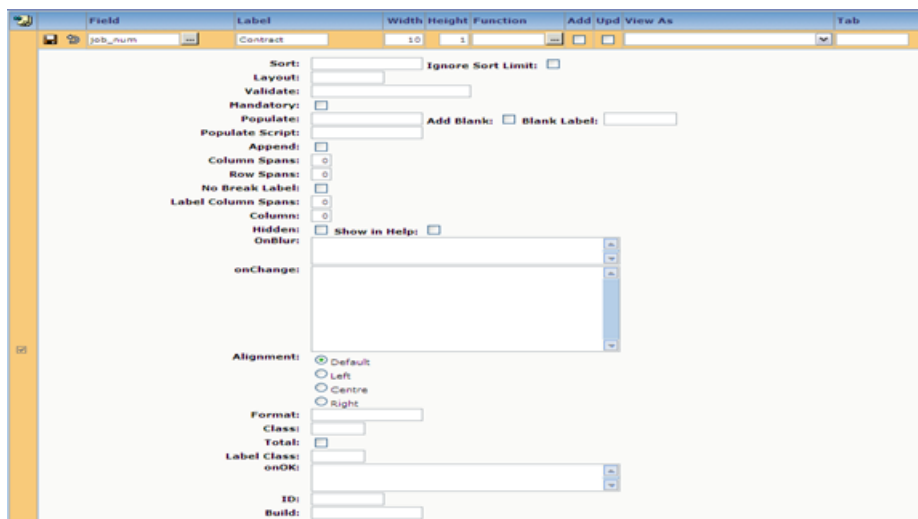
- Field Labels:** A table with columns: Field, Label, Width, Height, Fun. The 'Label' column contains 'Contract Number'.
- Sort By:** A dropdown menu with 'Contract Number' selected.
- Link Function:** A button labeled 'Add Blank'.
- Context Form & Fields:** A form displaying details for 'Contract 1000' at 'Oxford Road Office, London O1'.
- Body Span Form:** A form displaying details for 'Contract 1001' at 'Holiday Inn Slough'.
- Body Detail Form:** A form displaying details for 'Contract 1002' at 'Terminal 5 retail fitout'.
- Body Form:** A table listing various contracts with columns: Contract ID, Name, Location, Group, Type, Costs ID, Revenue ID, Profit ID, Margin ID.
- Totals Form:** A form displaying summary statistics for 'Contract Costs'.
- Body Selector Form:** A form with 'From' and 'To' date pickers.
- Combo Box / Populates:** A dropdown menu for 'Matches'.
- FunctionX:** A note explaining that icons are created by naming a function with 'X' (e.g., Excel/Print/Showall icons).
- Browse Filter Tab / Advanced Filter s:** A note explaining that filters are used (e.g., GE, LE, MA).





The Page Field Tab screen will only display those fields which are configured to appear in the Form type applied in the filter. Ensure the correct Form Type filter is selected for the fields being maintained.

When creating a Browse/Enquiry function the fields should NOT be ticked as either Add or Update.



When working in Page Designer the Width field will be interpreted as a percentage of the screen area available.

A lookup is available as to which Fields are available to add onto the Form, or refer to the Database Enquiry



Once the Function and Page are complete (and the function has been added to an appropriate menu) the Function can then be run.

#### 7.7.5.2 Page Forms - Add

This is similar to a preadd form in that it is required before an add. The difference is that the Add Type form appears in the button bar before the Add Button. There is no structure to the form, the fields are simply placed side by side before the add button.

The example where this will be first used is the adding of a PO line. There are a small number of different types of line that can be added. The contents of the body change depending on the type of line being added. By having the add type field available in the URL that is passed to the add page, the page can be changed dynamically by looking at the value of the add type field.

Add type fields are prefixed addtype.

So, if pol\_type was specified in the addtype form then the URL would contain addtypepol\_type=XXX and this could be used to create the appropriate type of order line.

You must use view-as combo for add type fields.





### 7.7.5.3 Page Forms - Body

The Body defines the fields to appear on a browse line.

#### 7.7.5.4 Page Forms - Body Detail

A Body Detail form defines the fields to display in an extra frame below a browse line (using the More button).

From the main body line of a Page, it is then possible to dig down to further details on the item. Create a Form Type of Body Detail and then create some Body Detail fields.

When the Function is run, it now will have an option to dig deeper into the Body Detail of each of the Body fields.

It is not possible to dig down into transactions using Body Detail – this is achieved by using a separate enquiry.

### 7.7.5.5 Page Forms - Body Selector

A Body Selector contains the select elements (combo fields) which refresh the page when changed, passing their value to the new page. This allows you to change the record set or contents of the records shown. The Body Selectors are pre-defined and will require certain tables to be assigned as Header Tables to ensure that the appropriate RSP's are available.

You can put a "view as" option on a body selector for the different combo types (Value, Value/Description, Description/Value or Description). The default is the value only.

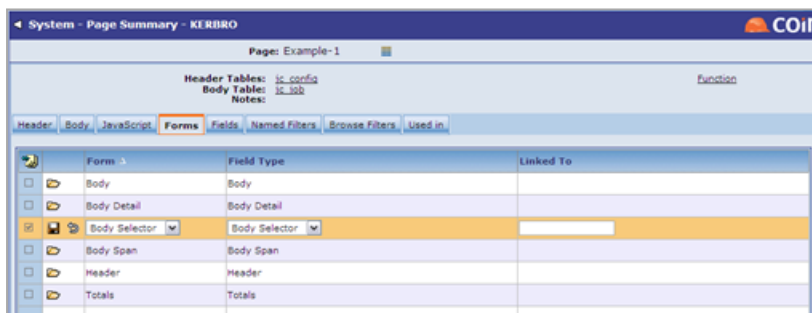
#### Body Selector Example

Contract Periods – available in jc\_job (job-rsp)

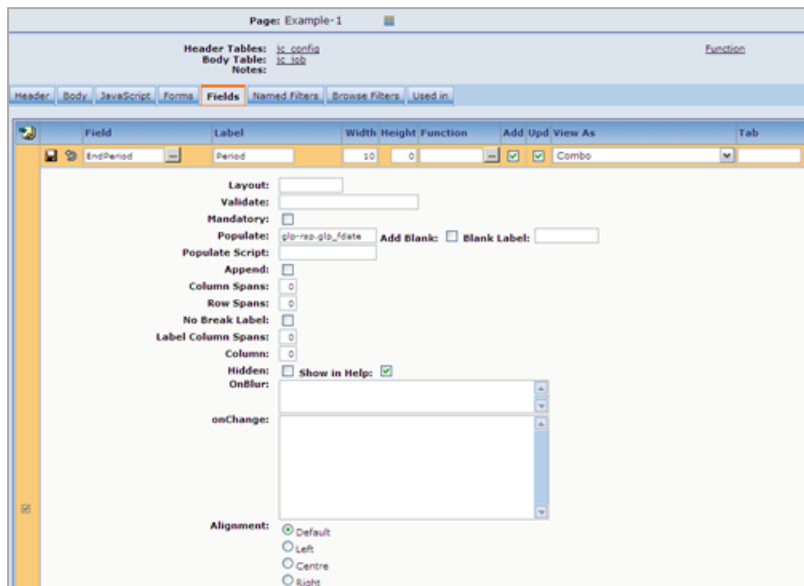
**EndPeriod** and **StartPeriod** Populate = glp-rsp.glp\_period

An example of this is to restrict values to a certain Financial Period by adding a Period Selection box in a Body Selector Field.

Create a Body Selector form on the page.



Add Page Field for Body Selector :



Define the Body Selector, for example, EndPeriod and give it a label. EndPeriod can be populated via glp-rsp.glp\_fdate and view-as set to Combo.

To the main Body of the Page add a RO\_ Field which requires parameters.

For example:

```
RO_CSSLValue^<CSSLValueIndex>|<PeriodType>[|<PeriodOffset>[|<FDate>
[|PhaseMasks]]]
```

- ^ Parameters Start
- <> Mandatory Parameter
- [|] Optional Parameters

An example of the Field Configuration is ...

```
RO_CSSLValue^2|TP|0|{EndPeriod}
```

Where {EndPeriod} will be populated from the Body Selector.

#### Populating body selectors from generic lookup tables

There are some generic 'Body Selector' fields which will enable you to be able to populate a body selector via the use of one of the generic lookup tables (ie co\_table,sy\_table or cr\_

table).

**Syntax**

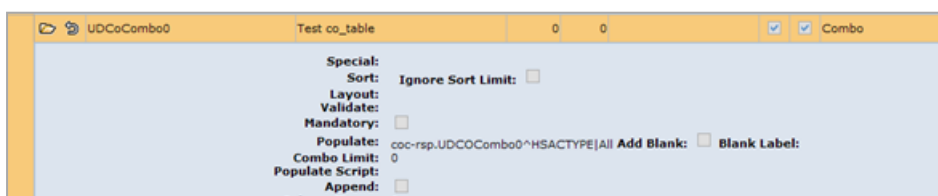
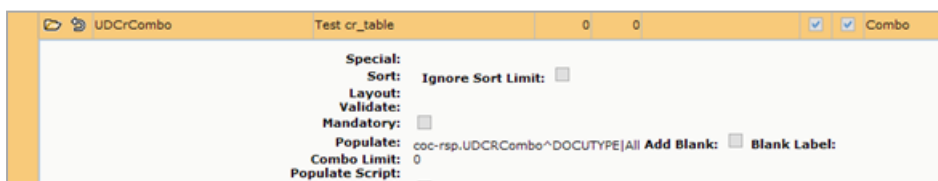
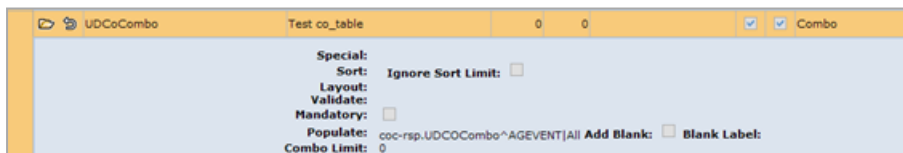
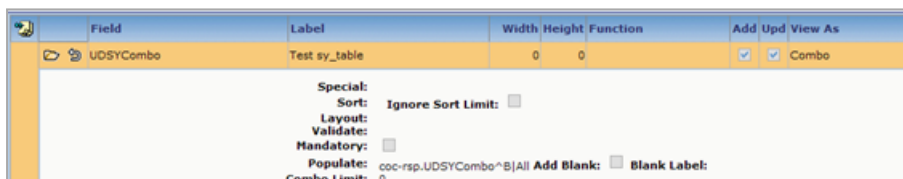
There is a field for looking at records held in :

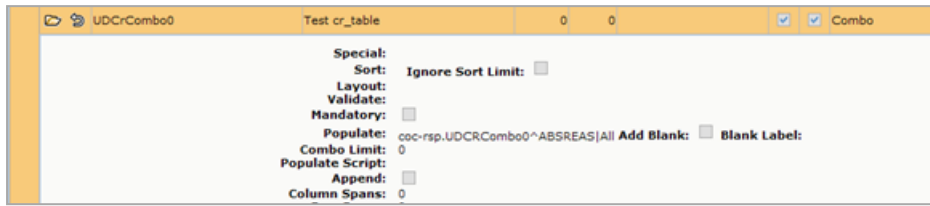
- `sy_table coc-rsp.UDSYCombo^{Name of sta_rtype}[[Name given to Blank record]]`
- `co_table coc-rsp.UDCOCombo^{Name of cta_rtype}[[Name given to Blank record]]`
- `cr_table coc-rsp.UDCRCombo^{Name of ctr_rtype}[[Name given to Blank record]]`

For situations where a company 0 is in use, additional options are available:

- `co_table coc-rsp.UDCOCombo0^{Name of cta_rtype}[[Name given to Blank record]]`
- `cr_table coc-rsp.UDCRCombo0^{Name of ctr_rtype}[[Name given to Blank record]]`

**Examples:**

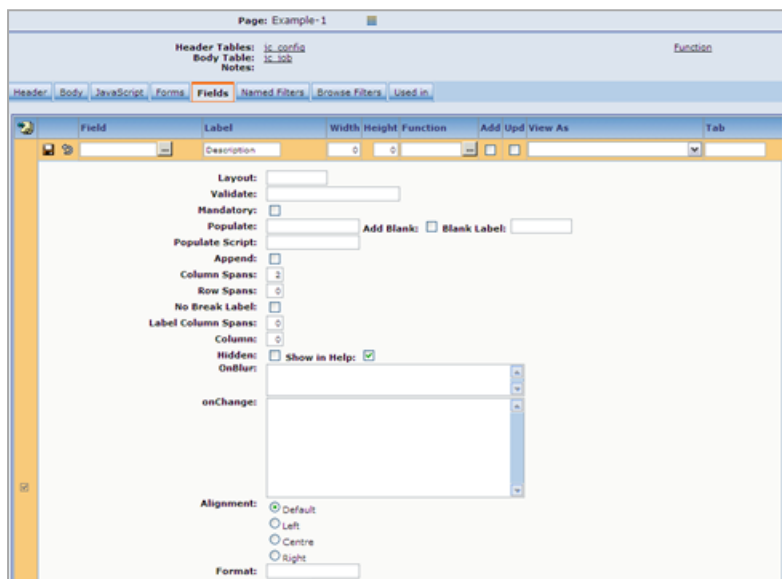




### 7.7.5.6 Page Forms - Body Span

A Body Span will provide additional title and column labels to the screen.

Add in a Body Span Form Type and then add Page Fields against the Body Span Type. Against the Body Span field enter a Label and in Column Span add how many columns this Label should extend over.



A summary style header will now appear on the Enquiry Screen.

Contract Status - Example Page Query - KERBRO				Values to Date	
Contract	Contract Name	Description	Contract Costs	Contract Costs (Period)	
<input checked="" type="checkbox"/>	1000	Oxford Road Office.	2,122,952.84	2,600.00	
<input type="checkbox"/>	1001	Holiday Inn Slough	3,605,354.45	0.00	
<input type="checkbox"/>	1002	Terminal 5 retail fitout	6,830,097.40	0.00	
<input type="checkbox"/>	1100	A406 Ealing underpass	854,830.22	0.00	
<input type="checkbox"/>	1101	Windsor Rail bridge replacemen	302,662.54	0.00	
<input type="checkbox"/>	1102	Eton River Thames bank rebuilding	555,629.23	0.00	
<input type="checkbox"/>	1103	Heathrow T4 hydrant fuel insta	387,821.36	0.00	
<input type="checkbox"/>	1104	M34 Northampton - Southampton.	21,870,051.32	1,300.00	
<input type="checkbox"/>	1111	Mxd.Development at Precinct 14	15,000.00	0.00	



#### 7.7.5.7 Page Forms - Body Update

A Body Detail defines the fields to update in an extra frame below a browse line.



#### 7.7.5.8 Page Forms - Context

The context appears immediately below the title bar.

It provides the context of the record you are viewing, with links to the parent record(s). It is non-updateable and is generated automatically (although you can override it).



#### 7.7.5.9 Page Forms - Detail

A Detail form defines the fields to display on a separate page.



#### 7.7.5.10 Page Forms - Footer

A summary form (containing header table information) after the body. It is the same as a header, only at the bottom.



#### 7.7.5.1 Page Forms - Header

A Header appears at the top of the page immediately underneath the page context area. It is intended to further or better describe the header record(s) of this page. It can also be used to show a summary page.

#### 7.7.5.12 Page Forms - Multiple Update

A Multiple Update form provides a separate form that allows you to update several fields on multiple (selected) records at the same time (using the Multi-update button). Do this by creating a multi-update form and adding the fields you want to update.

If the values on all records are the same then the default is set for the update and the update check box is ticked.

If any record values are different then the value is not set and neither is the check box to do the update.

#### 7.7.5.13 Page Forms - Record Header

A Record Header appears above tabs in Update or Detail. It allows you to put record context (as opposed to parent context) above the record you are updating. It can also include fields that are updateable. The record header appears above any tab folders that are specified and is therefore visible on each tab. This would be useful say on a maintenance screen where the code and description are put in the record header and the other fields in the tab section.

#### 7.7.5.14 Page Forms - Totals

Provides a totals form for a browse. If you total a non-integer/decimal field you will get a field count, otherwise you will get the sum of the value displayed.

Totals can also be placed at the bottom of the columns. On each field of a page section there is a field Total: . If ticked and you create a Total form, the total for the column will be placed in a total line at the end of the body section when the total button is pressed. Any fields added to the total form are then presented under this body total line. A record count is placed in the first column (selection/filter section).



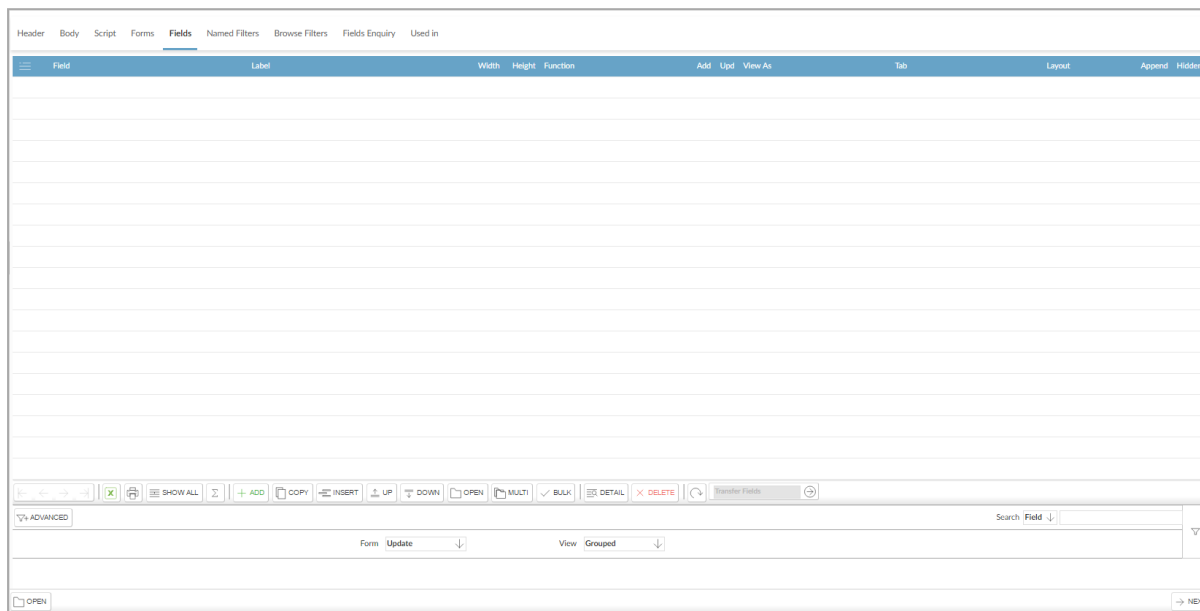
#### 7.7.5.15 Page Forms - Update

An Update form defines the fields to update on a separate page.



## 7.7.6 Page Fields Tab

This tab allows you to specify which fields appear on each of the forms on the page.



Page Selectors:

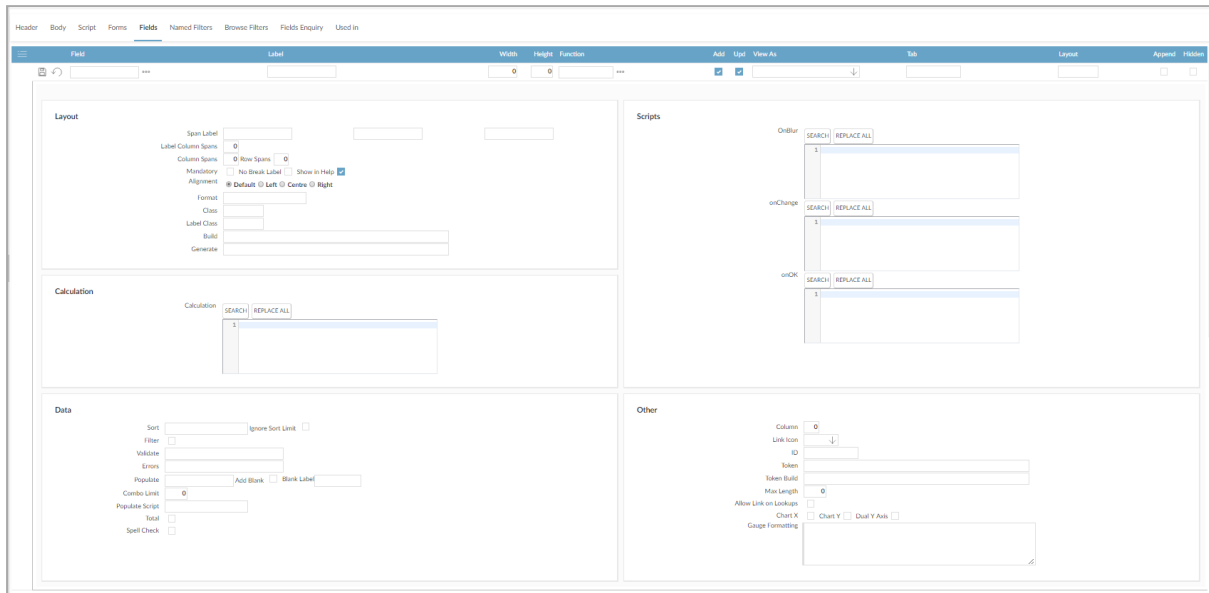
Field	Description
Form	Select the page form that you want to enter or update fields for.  Make sure that you have defined the appropriate page form using Page Form Maintenance.
View	When you click Open or view detail, the additional fields can be shown in a Linear form (Standard), or arranged in groups of like fields (Grouped)

Fields:

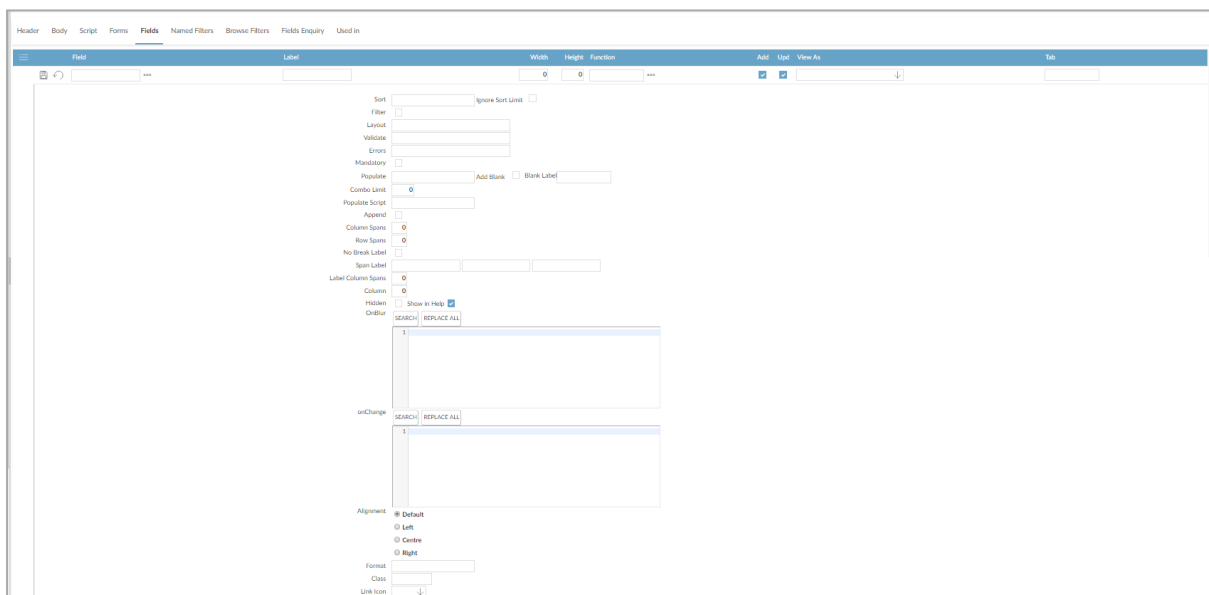
Field	Description
Field	The name of the field in the field set.
Label	Leave this blank to display text only (for example, a table heading).  The label for the field.  This will be either a column heading or a side label, depending on how the

Field	Description
	field is used.
Width	The column width as a percentage of the width of the page (for browsers).
Height	The height of the field.
Function	The code for a function to link to from this field (optional).
Add	Whether the field is available for update when adding a new record.
Upd	Whether the field is available for update when updating an existing record.
View As	How the information in the field is presented to the user.
	View As options
Tab	The function code for the tab this field is displayed on (if the page is divided into tabs).
Layout	The layout that this field is to be used in.
Append	Whether this field is appended to the previous field. <ul style="list-style-type: none"> <li>On Detail or Update pages, an appended field is displayed after the previous field, on the same line.</li> <li>On Body pages, an appended field is stacked below the previous field</li> </ul>
Hidden	Whether the field is hidden from the user when the screen is first displayed. <p>You can use this:</p> <ul style="list-style-type: none"> <li>To update a field, such as an internal reference, that the user does not need to see. Whatever is in the field gets saved and sent.</li> <li>To hide a field that is only required in certain circumstances. For example, an "amount liable to VAT" field which only applies to VAT lines. You could use a JavaScript function in the OnOK of another field to make this field visible.</li> </ul>

#### Additional Fields (Grouped View)



### Additional Fields (Standard View)



### Additional Fields:

Field	Description
Span Label	Span label level 1, Level 2 & Level 3
Label Column Spans	The number of columns the label for this field takes up.

Field	Description
	<p>If you put a label on an appended column-spanned field then the label is not shown. If you enter a number in the Label Column Spans field then the label is shown and will span that number of columns.</p>
Column Spans	<p>The number of columns this field takes up.</p>
	<p>On a web screen, fields are cells in a table, and each cell can span more than one column. The total number of columns depends on the line with the largest number of 'column spans' (including appended fields).</p>
Row Spans	<p>Number of rows to span.</p>
Mandatory	<p>Whether this is a required field (that is, it cannot be left blank during input).</p>
No Break Label	<p>Select to prevent a label from being split on to multiple lines</p>
Show in Help	<p>Whether the help frame includes help for this field. Clear this box, for example, if the field is never visible to the user.</p>
Alignment	<p>How the field is aligned. The default alignment depends on the data type of the field.</p>
Format	<p>A PROGRESS display format that overrides the standard formatting for the field.</p>
Class	<p>The standard style sheet class for this field.</p> <p>By default, the text is appended to the underlying class name of the cell. For example, "bold" could be appended to the standard classes "odd" and "even" to produce <code>&lt;TD class="oddbold"&gt;</code> and <code>&lt;TD class="evenbold"&gt;</code>. You will need to specify the formatting for these new styles in one of the cascading style sheets.</p> <p>To specify a class as an addition to the underlying class, add a dot before the class name. For example, <code>.coins_yellow</code> will be added to the standard classes to produce <code>&lt;TD class="odd coins_yellow"&gt;</code> and <code>&lt;TD class="even coins_yellow"&gt;</code> (the yellow background will be added to the standard "odd" or "even" class).</p> <p>This can contain <code>{}</code> to change the class programmatically.</p>
Label Class	<p>Text that is appended to the standard label class for this field.</p>
Build	<p>An RSP and name of a method to call to determine whether this field is shown on the form.</p>
	<p>If the return value is FALSE then the field is ignored and will not be present in the form. This is intended for use with parameters.</p>

Field	Description
Generate	<p>This allows you to include fields from other sources at this point in the page. It can be either:</p> <ul style="list-style-type: none"> <li>• A form from another page, in which case all the fields from that form will be included. Specify <code>page.form</code>; for example, <code>%WPLIAIN001.UPDATE</code>.</li> <li>• A program and method to programmatically generate the fields. For example, <code>glfgjh01.generateTokens</code>.</li> </ul>
OnBlur	<p>JavaScript functions to call when the user leaves this field (whether or not the value changes).</p> <p>The functions need to be available to the page; for example, you can specify <code>.js</code> files in the Script field of the page header.</p>
onChange	<p>JavaScript functions to call when the user changes the value of this field. These functions are also run when the page first loads.</p> <p>The functions need to be available to the page; for example, you can specify <code>.js</code> files in the Script field of the page header.</p> <p>This option is mutually exclusive with the validate field option and will be ignored in this case.</p>
onOK	<p>The name of a JavaScript function to run when the field is validated. For example, when you enter a contract number the contract name is displayed alongside. Multiple scripts can be executed by separating with ";".</p> <p>Two useful JavaScript functions to call are:</p> <ul style="list-style-type: none"> <li>• <code>setReturnInput(&lt;target&gt;,&lt;source&gt;);</code> input fields</li> <li>• <code>setReturnField(&lt;target&gt;,&lt;source&gt;);</code> display only fields</li> </ul> <p>The source of the data is the return variable in the validation procedure. If source is not specified then the target name is used for the source.</p>
Calculation	<p>Calculation to display</p>
Sort	<p>The field that is used as the sort index when records are sorted by this column. Normally, this will be the name of the field itself, followed by <code>+</code>. If this is blank, the field is not sortable.</p>
Ignore Sort Limit	<p>Select if the browse sort limit should be ignored for this column sort. You should only select this if there is an index to support this sort order for very large record sets.</p>

Field	Description
Filter	<p>Whether to include this field on the browse filter automatically if the field is shown on the browse.</p> <p>This is particularly useful where column sets are in use - if the field is shown on the browse, it is added to the simple filter drop-down list; if the field is not shown, it is not included in the list.</p>
Validate	<p>How to validate this field. This can be either:</p> <ul style="list-style-type: none"> <li>• An RSP name; this will run the ValidateField method from this RSP.</li> <li>• An RVP name and the checks to apply. For example, jcvjob.l2,* will run the standard checks in jcvjob.p, except check 2.</li> </ul>
Errors	Which error messages are associated with this field
Populate	Enter the RSP, or RSP and procedure, to be used to populate this field.
Add Blank	When ticked, (if this field is a combo) Add a blank option to the list.
	Populate methods (or syu008.i) return valid records, and you select whether a blank is also required.
Blank Label	If Add Blank is ticked, this is the label to use for the blank option (for example, "All"; the default is "None").
Combo Limit	The maximum number of values to be shown in a combo. If this number is exceeded then the combo will be replaced by a fill in.
Populate Script	Script to run instead of the populate method to populate the data (for example, to copy options from a similar object on the page).
Total	If you want this field to be totalled (on a browse), tick the box. You also need to create a Total form for the page.
Spell Check	Whether spell check should be applied to this field.
	Applicable to character fill in fields only. Simple editor fields have spell check enabled by default.
Special	Program used to produce HTML for the field (if using View As = Special)
Column	<p>A column number. Fields with the same number will be put in the same column.</p> <p>This allows you to format simple pages. However, using Append and Column Spans is a better way to lay out columns.</p>
Link Icon	Specify what link icon is to be shown alongside the link (if used).
ID	An ID for the field.

Field	Description
	<p>This is used in two ways:</p> <ul style="list-style-type: none"> <li>• For blank fields (that is, labels only), to specify the name of the field that is used to display help for the label. This is useful for column labels.</li> <li>• For input fields where a program (such as a CUI report) requires a specific fieldname to be passed back, but it is better to use a different actual field name to allow for lookups and help. For example, the report expects rs_dates__1, but you are using the field RS_glp_fdate__1; put rs_dates__1 in the ID.</li> </ul> <p>This can also take a second entry (comma-separated) which is the name of a field that is used to display help. This is used to override the help for some fields on standard pages.</p>
Token	<p>The token ID of the field so that it can be selected in a column set.</p> <p>If blank then the field name is used. If ! then the field will be excluded from a column set.</p>
Token Build	<p>A condition method which if returns true will build the token ID to allow column sets to use this field.</p>
Max Length	<p>The maximum length allowed for input to this field. Maximum Length: 3 characters</p>
Allow Link on Lookups	<p>Links are normally disabled during lookups. If you select this option then the link will remain active if this page is used in a lookup.</p>
Chart X	<p>Whether this field is used as the X axis on charts. If you select multiple X axes, then the values will be concatenated.</p>
Chart Y	<p>Whether this field used as a Y series on charts. You can select multiple Y series and show on a multi-series chart.</p>
Dual Y Axis	<p>Whether this value should be plotted on the second Y axis on the chart.</p>
Gauge Formatting	<p>Extra formatting which will be applied if this field is viewed as a gauge.</p>

#### 7.7.6.1 Page Fields - View As Options

##### **blank**

Allows you to display the field as an input field.

##### **Blank When Zero**

Allows you to display the value of a numeric field, and display a blank when the value is zero.

##### **Checkbox**

Allows you to display a logical field as a checkbox.

##### **Code with tooltip**

Allows you to display the code value as normal but will provide a tooltip when you hover over the code showing the description. This uses the combo populate technology, so you must have a `getValue<Field>` available in the appropriate RSP. This may not work very well with updates or with linked fields but might be useful at other times.

##### **Combo**

Provides a drop-down from which you can select the value of the field.

To specify how the combo is populated, in the Populate field is an RSP name and optionally the name of the populate procedure to call. e.g. `job-rsp` or `job-rsp.job_complete`.

You can populate a combo with a user defined list of options. The syntax for the populate is as follows:-

```
sys002.combo^A,Desc A,B,Desc B,C, Desc C
```

As a default the combo will display in alphabetical order. To override this behaviour you can specify the order by inserting Pipe symbol prior to the Description. For example:-

```
sys002.combo^C,|Desc C,A,|Desc A,B,|DescB
```

If the field is a user-defined code, show the value and description. If the field is a system-generated code, show the description only.

##### **Variants:**

- Combo
- Combo Description/Value



- Combo Value Only
- Combo Value/Description

You can populate Combo-boxes using client-side code.

For example, on the Facilities Management - Add Job screen (page %WFM1010BSWJ), changing the Major Category will change the possible selections of Minor Category and Response.

There is a JavaScript file `populateCombo.js` which contains functions for populating description/value and value only combo boxes. Various data is sent to the page using a couple of Send Data procedures. This data is then manipulated by the function `majcat_onchange` in `FMstuff.js` which calls `populateCombo`.

If you have too many records in a combo then the combo as an input device becomes unusable. Also if you have lots of records then eventually you will break the string limit in PROGRESS and the input field will fail. One workaround to this is to put two fields, one as an input without combo and a display field with combo. There is now a count limit to all combos such that if the count of options in a combo exceeds 400 then the combo is now shown and instead a fill in will be shown instead. There is now an added option on the section field to allow you to specify the limit to be shown in the combo. The Combo Limit field in under the Populate field. If this is not specified it will default of 400 will be used.

#### **Disabled Field**

Allows you to display a field that is always disabled, but that you can put values into using JavaScript. For example, a total field on a form that is the sum of some other fields.

#### **Disabled with Lookup**

Allows you to display a field as a disabled field and provide a lookup button alongside to populate it.

This is equivalent to a server-side combo.

For example, people fields on a HS visitor record.

It could also be used to select a company for a PO although this is currently done with an enabled field and field validation.

You will need to set up a lookup for the field in the normal way and return an appropriate description. Also using the lookup you can return an additional code in to other fields on the page (hidden or otherwise); for example, the internal reference of the field.

#### Editor

Allows you to display the field as an editor box. The box has vertical scroll-bars if required.

Avoid displaying Editor contents in a browse (Body form). Use a Body Update or Update form.

#### Email

This is similar to View As Link except that it automatically adds on the mailto: to the link so that it links to your default email client.

#### File

Allows you to update a field as a filename.

In Update, the field has a Browse button next to it, which opens a Windows “Choose file” dialogue box. The file you select is uploaded to the server and placed in the upload directory for further processing.

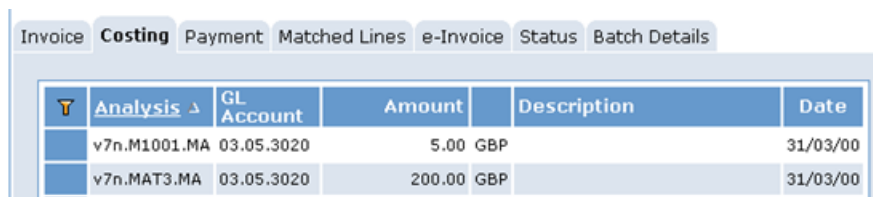
#### Grid Element

A grid element is a cell in a matrix.



#### Inline Frame

An inline frame is a separate window and can contain a browse page which can using paging techniques to navigate a large number of records.



Analysis	GL Account	Amount	Description	Date
v7n.M1001.MA	03.05.3020	5.00 GBP		31/03/00
v7n.MAT3.MA	03.05.3020	200.00 GBP		31/03/00

In the Function field, specify the function that is to be included at that point in the header page. Functions/Pages of this type should be named like a normal browse type e.g.

%WmmBttt. If you specify a label then the frame will be labelled. If you don't then the frame will take up the complete width of the header.

#### Link

Allows you to display a field as a Link.

This assumes the field contains a valid URL or Filename that can be linked to.

In Update the field is presented as a fill in for entry.

In display the field becomes a link to the document specified.

You can specify the link text to be different to the link itself. For example, if you have a link to a document F:\documents\file.doc, that would by default appear as the text for the link. If you enter something in the Populate: field for this field, then that is used instead. You can also use {} to replace with data in the Populate: field.

#### List Frame

(Used on a Header form) A list is a non-paged set of records. You can control the size of the containing frame by specifying a height on the field -this can result in a scrollbar being added (if you leave the height blank, the list grows to display all records). However, all the records are downloaded in to the header page (beware the volume of data). In the Function field, specify the function that is to be included at that point in the header page. Pages of this type should be named %WmmLttt where mm is the module, L (for List) and ttt is the table TLA. If you specify a label then the list will be labelled. If you don't then the list will take up the complete width of the header.

VAT Analysis:	VAT Code	VAT Rate	Description	Goods Amount	VAT Amount	
	1	17.50	Standard Rate	975.00	170.63	GBP

#### Multiple Selection

You can view comma-separated lists as multiple check boxes.

e.g. cim\_typelist on ci\_company is a list of company types.

You setup populate/getvalue functions in the RSP exactly the same as a combo. The result is a set of checkboxes, one for each of the options in your populate function, ticked if the field you are updating contains the code in its list.

<input type="checkbox"/> Cable Companies	<input type="checkbox"/> Partner 2
<input checked="" type="checkbox"/> Client	<input type="checkbox"/> Partner 1
<input type="checkbox"/> Consolidator	<input type="checkbox"/> Partner 3
<input type="checkbox"/> Consultant Engineer	<input type="checkbox"/> Partner 4
<input checked="" type="checkbox"/> Customer	<input type="checkbox"/> Plant Hirer
<input type="checkbox"/> Electricity Company	<input type="checkbox"/> Quantity Surveyor test
<b>Types:</b> <input checked="" type="checkbox"/> Emergency Callout Companies	<input type="checkbox"/> Sewerage Companies
<input type="checkbox"/> Estate Agent	<input type="checkbox"/> Solicitors
<input type="checkbox"/> Financial Advisors	<input checked="" type="checkbox"/> Subcontractor
<input type="checkbox"/> Gas Supply Company	<input checked="" type="checkbox"/> Supplier
<input type="checkbox"/> Guarantee Schemes	<input type="checkbox"/> Telecommunication Companies
<input type="checkbox"/> Hospitals	<input type="checkbox"/> Training Provider
<input type="checkbox"/> Inspector	

The display version shows only the ticked options.

<b>Types:</b>	<input checked="" type="checkbox"/> Client	<input checked="" type="checkbox"/> Subcontractor
	<input checked="" type="checkbox"/> Customer	<input checked="" type="checkbox"/> Supplier
	<input checked="" type="checkbox"/> Emergency Callout Companies	

Variants:

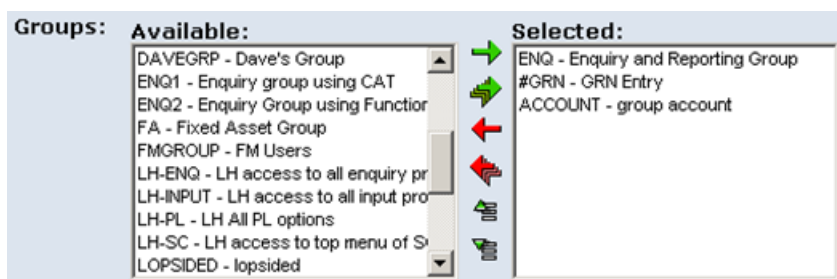
- Multi-Select
- Multi-Select Value/Description
- Multi-Select (2 Columns) - displays as two columns of checkboxes.
- Multi-Select Value/Description (2 Columns)

**No Break**

No Break will ensure that the field is not broken up when the available space is too small. This is done automatically for integers and decimals but you can use this view as option for other fields if required. Particularly useful for codes that might contain hyphens.

**Ordered List**

Allows you to update ordered list fields with two separate lists with add, delete, move up and move down buttons.



Set view-as on the field to Ordered List (or one of the value and description options). You will need a populate method (same as for combos, selections, etc.) and you will be presented with a list of unused codes in a left selection list, and the selected codes show in a right selection list.

The display version shows a comma-separated list of the selected codes.

**Groups:** 9PO-SCAGREE,root,#BUYER,#CIS,CLERKS

Look at %WSYBSUR (User Maintenance - Groups Tab) for an example.

Variants:

- Ordered List
- Ordered List Description/Value
- Ordered List Value Only
- Ordered List Value/Description

**Password**

Allows you to update and view a field as "password".

This will display \*\*\* in the place of the actual field.

**Password:**

**Picture**

View-as picture will present an image on screen when in display/read only mode and a fill in field (for the file/URL reference) when in update mode.

See the picture on the HR Person record for an example.

### Preformatted

Allows you to display a field as preformatted.

This will add `<PRE>` `</PRE>` around the HTML for this field. It assumes the data will be formatted in fixed width columns.

Select Preformatted on the view as field on the page field maintenance.

This view of data is provided to allow some existing code to be used in a web page. You should not use this type of View As except in this circumstance.

### Radio Set

Allows you to display input fields as radio sets.



Set it up like a combo (that is, with a populate option) and change the view as option to radio set. You will get a vertical radio set. Display will be the same as a combo i.e. the description of the selected option will be shown.

**Alignment:** Default

### Selection

Allows you to display data in a multi-selection list.

These are equivalent to the combo selections except that they accept multi selections and return a comma-separated list as a result.

The use of these is not recommended for updating records as it is too easy to click on the list and delete the data that is already selected. For this you should use multi select checkboxes.

These can be used for forms, for example report selection or processing selection where the initial value is probably blank.

You need to specify a populate method in the same way as you do for combos.

Variants:

- Selection
- Selection Description/Value
- Selection Value Only
- Selection Value/Description

**Sorting Combos, Selection List, Multi Selects and Ordered Lists**

These are sorted automatically in to the order that is selected in the view as.

For example, if you select view as combo value/description then the sort will be on the value.

If you select view as combo then the sort will be on the description.

If you want to override this (for example, GL Periods are sorted in date order not description order), then specify | as the first character of the description field. This can be specified in the syu008.i entry as &fixOrder=|.

## 7.7.7 Named Filters

Named Filters provide a drop-down selector that allows the user to select records that match a pre-defined query (for example: all, posted or unposted batches). Usually Named filters extend the Body Query specified on the page decreasing the number of output records.

### 7.7.7.1 Replacing body query

Also you can specify a named filter that replaces the Body Query on the page rather than adding to it (it is possible to change body table as well). If the query you set on the Named Filter begins "FOR " then it will replace entirely the query specified in the query on the page section. This can allow you to use different tables in your query. Example where this came up was a list of suppliers where the normal query on the page section was:

```
FOR EACH ap_vendor WHERE ap_vendor.kco = {kco}
```

We can then add a named filter with a query `avm_sclab = YES` to give a filter on those that supply SC labour and another with a query

```
FOR EACH ap_vendor WHERE ap_vendor.kco = {kco}, FIRST ap_invoice OF ap_vendor  
WHERE ap_invoice.ain_entry = 1
```

which will give all suppliers where they have an invoice that is costed to a contract.

### 7.7.7.2 Linked Named Filters

You can link named filters. If you have standard filters that need to be applied to a number of different but similar pages you can now link (as well as the forms which has been available for some time) the named filters. There are two fields on the page section maintenance to allow you to specify the page section to use to retrieve the named filters. If you leave them blank then it will use the filters defined on the current page section.

### 7.7.7.3 Multiple Named Filters

It is possible to have multiple named filters that are applied to the query together. If you select the New field on the named filter then this will start a new named filter combo from that named filter onwards. The query condition on the first (existing) set of filters can start with a replacement for the main body query FOR EACH. Subsequent named filters can only append additional query conditions to this. In this way you can build a matrix of filters that can be applied separately or together rather than having a large list of all combinations in the single combo as we had to previously do.



#### 7.7.7.4 Named Filter Query Conditions

You can now override the query condition function on the body query on a named filter. There is a new field Query Condition on the named filter. If this is non blank then it will be used in preference to the one specified in stn\_qcondition on the page section. If you are using multiple named filter combos then the query condition can only be applied to filters in the first combo which are the root query to be used (same as overriding the FOR EACH in the query)

#### 7.7.7.5 Using Named Filters to define what fields are built on a form

You can now use Named Filters in conjunction with Build Conditions to determine what fields are built on a form.

**Parameters:**

Mandatory:	
<b>Name</b>	The name of your filter how it'd appear in the combo-box ("All" to show all entries)
Optional	
<b>Table</b>	The name(code) of the table for the query
<b>Build</b>	RSP-function to build field
<b>Query String</b>	appended to your query

,

## 7.7.8 Browse Filters

Browse filters allow the user to determine additional conditions to query to show a smaller number of records. Browse filter is shown as the "Advanced filter" at the bottom of Browse. You can now specify a data type and format on advanced filter fields. If specified then standard input data type formatting will take place.

For example :- dates will be expanded, decimal points, commas added etc.

If you leave the data type blank then no formatting will take place as before.

### 7.7.8.1 Parameters:

Mandatory:	
Field	The field code for your query
Label	The name of your filter how it'd appear in the page
Type	Specify the conditions to execute the query (e.g. LE, GE, MA, etc.)
Optional	
View as	Type of the field (e.g. Combo, Checkbox , etc.)
Populate	Specify rsp, containing populate for Combo and MultiSelect fields
Build	RSP-function to build field
Query	String, appended to your query
Lookup Field	lookupCode(?)
Data Type	Type of output
Format	Format of output

## 7.8 Report Designer

### 7.8.1 Key Features

- Copy and amend standard reports
- Build simple and complex reports
- Use simple and complex calculations
- Use conditional formatting
- Use graphs, graphical displays, KPI reporting and trending
- Display/update user defined data including report stores
- Use of Alerts to send Tasks or Emails
- Create automated report packs and distribution
- Create Data Marts
- Integrate with all generic reporting features
- Integrate with Workflow
- Log Files

The Report Designer can be used to maintain complex reports, inline reports, report packs and container reports.

## 7.8.2 Page Summary

Create a Page to be associated with the function. This is not the layout of the report but the web page to be displayed when running the report - that is, the selection criteria (the page will include 'Output Options' by default in addition to any other selection criteria defined).

As with browse, create the Page, using the same code as the calling Function. However, with reports there are additional fields which are populated to provide functionality (typically predefined Report Selection criteria).

In the Header section of the Page, define a FSP (Form Service Procedure) for the Body Table to be reported on; this will provide certain tools and functionality to the report such as standard selection criteria. *7.6, Forms Service Procedures and Report Selection Generates*

There are no other Body details required on the webpage as this is just the selection page for the report (the actual detail; fields, columns etc. are maintained in the Report Section).

Click Save to save the Page and move on to Forms and Filters.

Although the Page is to be associated with a report it is necessary to create an Update Form in the Page to allow the user running the report to enter their appropriate selection criteria.

Once the Update Form has been created, then add the fields. If standard selection pages are to be used (see FSP's above) add a blank field to the page and ensure that the GENERATE field points to the correct selectionGenerate in the Form Service Procedure.

Fields defined for pages for report selection need to be flagged as Add and Update and must be associated with an 'Update' Form to allow entry of selection criteria.

Each field can be assigned a Tab - the selection will appear on that Tab (The Tab must exist as a function, by standard convention the table is named funcT where func is the function/page/report code and T designates tab. See additional information on naming conventions).

Although Named Filters and Browse Filters appear as available in Reports there are only used in COINS OA for Page Design

.

### 7.8.3 Report Designer\_Using RS Fields

In addition to the standard Selection Generates available in the FSP's it is also possible to pass other selection criteria to a report, either to the report query or to any parameter driven (RO\_) fields used on the Report.

The RS\_ Fields are specified on the ?Update? Field portion of the Page Section for the report. To create a Report Selection field add the RS\_ prefix and selection type suffixes (?FROM? = \_\_1 , ?TO? = \_\_2 and ?MATCHES? = \_\_3) to the field required for selection.

Example:

To filter for ?Matches? on the ?job\_group? use the following:

RS\_jgr\_group\_\_3

Filtering the same field on a ?FROM ? TO? basis use:

RS\_jgr\_group\_\_1 and RS\_jgr\_group\_\_2

Reference these RS\_fields on the selection criteria of the report query, when doing this the program will replace the {RS\_fields} portion of the query with the actual values the user entered at runtime.

## 7.8.4 Report Designer\_Report Forms

Below are the Report sections available for use.

Fixed Title (generated automatically)		The Report Title and Company
Title		3 lines of text to show the key ranges.
Header 1 Form		Header for sort level 1
Header 1 Columns		Header for sort level 1 - allows placement of data according to the columns set up in Bo
?		
Header 10 Form		Header for sort level 10
Header 10 Columns		Header for sort level 10
Body Block	Body Header Form	Form placed above every line of the body.
	Body Header Columns	Line of data placed above every body line; fields align with columns in body line
	Body	Repeating detail line with columns and column headers
	Body Columns	Line of data placed after every body line; fields align with columns in body line
	Body Form	Multi-line form (stacked). Appears after each body line if used with a body form.
Footer 10 Columns		Footer for sort level 10. Allows placement of data according to the columns set up in Bo
Footer 10 Form		Footer for sort level 10.
?		
Footer 1 Columns		Footer for sort level 1.
Footer 1 Form		Footer for sort level 1.
Total Columns		"Grand Totals" for the report (according to the columns set up in Body)
Total Form		"Grand Totals" for the report
Report Form		Information to always appear at end of report - irrespective of data returned.
Fixed Footer (generated automatically)		"Printed using?" and page number

The Form sections should be added as required, as a minimum a Body section should be created.

Fields are then assigned to the Body as required. The fields can be extracted from the Database Enquiry depending on the query using the same principles as when setting up Pages for browsing and enquiries.

## 7.8.5 Headers and Footers

Multiple Headers and Footers are associated with the Sort options assigned to the Report Section. If Sort options have been set, simply click the Total box in Report Field Maintenance and COINS OA will define the appropriate subtotals and totals in relation to the sort options defined.

As long as header and footer forms have been defined for the report, to correspond to the sort levels, COINS will populate these automatically with the fields the report is sorting by. However, it is necessary to specify labels and descriptions for these fields.

To provide column totals, add a "Total Columns" form (for report run totals) or a "Footer Columns" form (for subtotals); again, no fields are required on the form; on the field records for the fields to be totalled, tick the Total box.

The contents of the header and footer forms can be overridden by adding fields manually.

## 7.8.6 Default Report Labels

Default report labels are used to populate the automagic behaviour behind the Header X Forms and Footer X Columns. They provide the Label Name and the Field List which appear in the standard Headers and Footers which appear on Reports by Sort Levels. It is important to remember that changes to this table will affect ALL reports and not just a single report.

If you would like to override this automagic behaviour on any report you can always add the fields as required to the relevant form on the bespoke report. (i.e. as soon as you add fields to a Header Form or Footer Columns you override the automagic behaviour). This is not always desirable as this means that you would have to keep your accumulated totals manually to display in the footer columns - It is probably more desirable to maintain the use of the Footer Columns and to also make use of the additional forms available at all levels do any additional calculations or display further information.

For example for every Footer X Column there is an equivalent Footer X Form which will appear directly beneath this form.

Footer Forms will automatically display information details one underneath the other so it is often necessary to append records to appear across the page.



## 7.9 Data Display Colour Ranges

Configuring Colour Ranges for use within COINS OA Reporting simply enter the code for the Range and then enter a comma separated list containing the ranges and classes for this code.

Data Display Colour Ranges are Company Specific and must be configured in every company in which a report may be used.

To set up the colours and ranges that COINS uses to display gauges and to highlight 'score' fields such as performance ratings :

Code	<p>The code for the data display type.</p> <p>For gauges, this code must be entered as the value of the Populate field for the field to be displayed.</p> <p>For 'score' fields, this code must be passed as the first parameter to the {co_config.RO_rangeClass} method in the Class field (the score field/score value being compared should be the second parameter).</p>
Description	<p>The ranges and colours for this code.</p> <p>These are of the form number,colour,[number,colour,] ... number, where number represents the value at which the colour display changes, and colour represents the colour used for displaying values between the two numbers on either side of it.</p> <p>For gauges, colour is a hexadecimal RGB colour code. For example, 0,#00FF00,3,#FFFF00,7,#FF0000,10 gives the range 0 1 2 3 4 5 6 7 8 9 10.</p> <p>For 'score' fields, colour is a cascading style sheet class defined in the user.css file.</p>

## 7.10 Alternate Line Shading on Forms and Reports

This guide provides instructions for COINS Business Intelligence and Reporting designers on how to shade alternate lines of PDF reports and forms.

It is expected that the user performing the tasks outline in this How-To guide has completed COINS Business Intelligence and Reporting training.

Alternate line shading allows the report reader to associate right most totalling or display columns with the detail on the left of the page as shown below:

<b>Revenue</b>											
<b>Client Fees:</b>											
0000 Client Fees	0	0	0	0	0	0	0	0	0	0	0
0000 Client Fees - Fee	0	0	0	0	0	0	0	0	0	0	0
0000 Client Fees - Fee	0	0	0	0	0	0	0	0	0	0	0
0000 Client Fees - Fee	0	0	0	0	0	0	0	0	0	0	0
0000 Client Fees - Fee	0	0	0	0	0	0	0	0	0	0	0
<b>Total Client Fees:</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>
<b>Other Income:</b>											
0000 Other Income	0	0	0	0	0	0	0	0	0	0	0
0000 Other Income - Fee	0	0	0	0	0	0	0	0	0	0	0
0000 Other Income - Fee	0	0	0	0	0	0	0	0	0	0	0
0000 Other Income - Fee	0	0	0	0	0	0	0	0	0	0	0
0000 Other Income - Fee	0	0	0	0	0	0	0	0	0	0	0
0000 Other Income - Fee	0	0	0	0	0	0	0	0	0	0	0
0000 Other Income - Fee	0	0	0	0	0	0	0	0	0	0	0
0000 Other Income - Fee	0	0	0	0	0	0	0	0	0	0	0
0000 Other Income - Fee	0	0	0	0	0	0	0	0	0	0	0
0000 Other Income - Fee	0	0	0	0	0	0	0	0	0	0	0
<b>Total Other Income:</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>

Always perform all implementation and testing of report and/or form alterations in an up to date, non-live environment before attempting to migrate your changes into the LIVE environment.

When migrating your changes to the LIVE environment it is recommended that you have a working, complete backup of the LIVE environment.



### 7.10.1 Prerequisites

The user performing the tasks in this “How-To” must have:

- COINS Business Intelligence and Reporting training
- A COINS designer license allocated to their COINS user account

## 7.10.2 Task Summary

The tasks required to implement alternate line shading in COINS reports and forms are as follows:

Create report field styles

Applying to forms (optional)

Applying to reports (optional)

The tasks are to be performed in the order outlined above. Each task is broken down into a series of steps that will need to be completed.

Some tasks are identified as optional. Whether or not a task is optional will depend on your specific COINS installation.

Please contact COINS support if you require any clarification of tasks contained within this "How-To" guide.

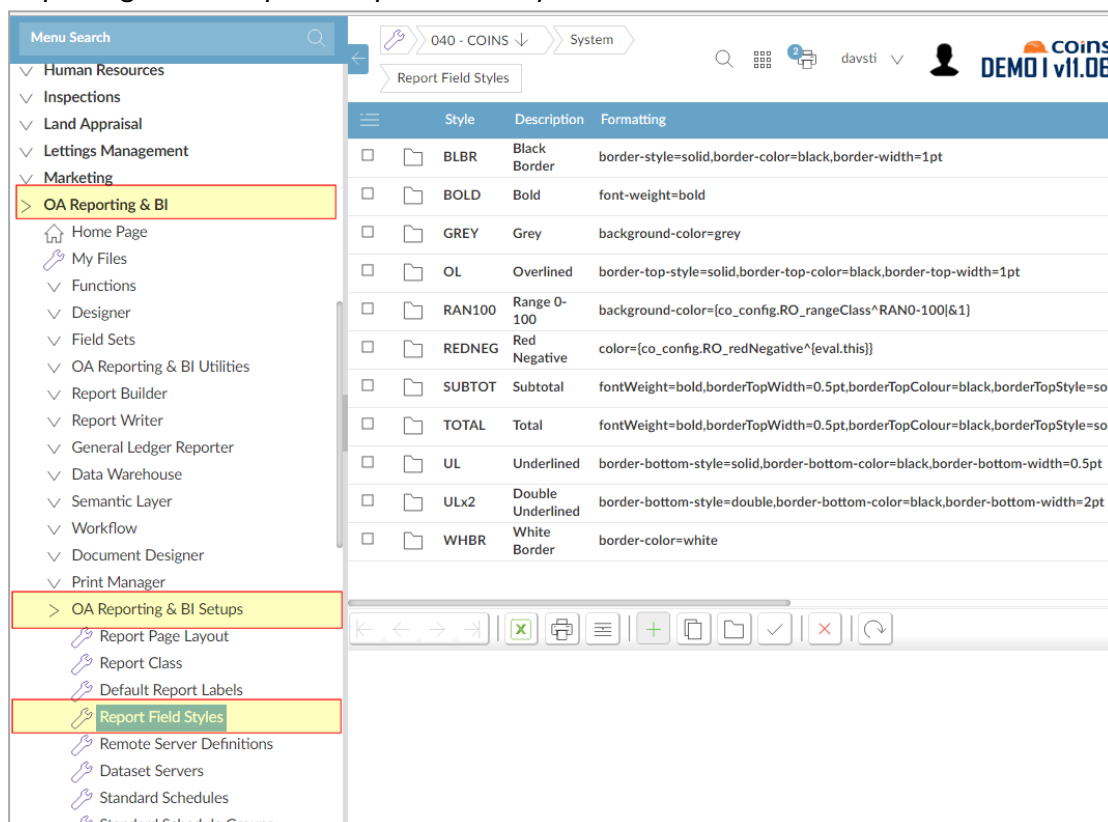
### 7.10.3 Create report field styles

The first task is to create a set of report field styles that allow reports and forms to invoke the field shading. The number of field styles that will be required will depend on the COINS installation and the number of existing field styles being used.

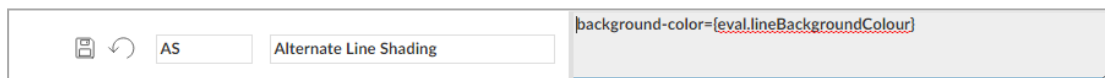
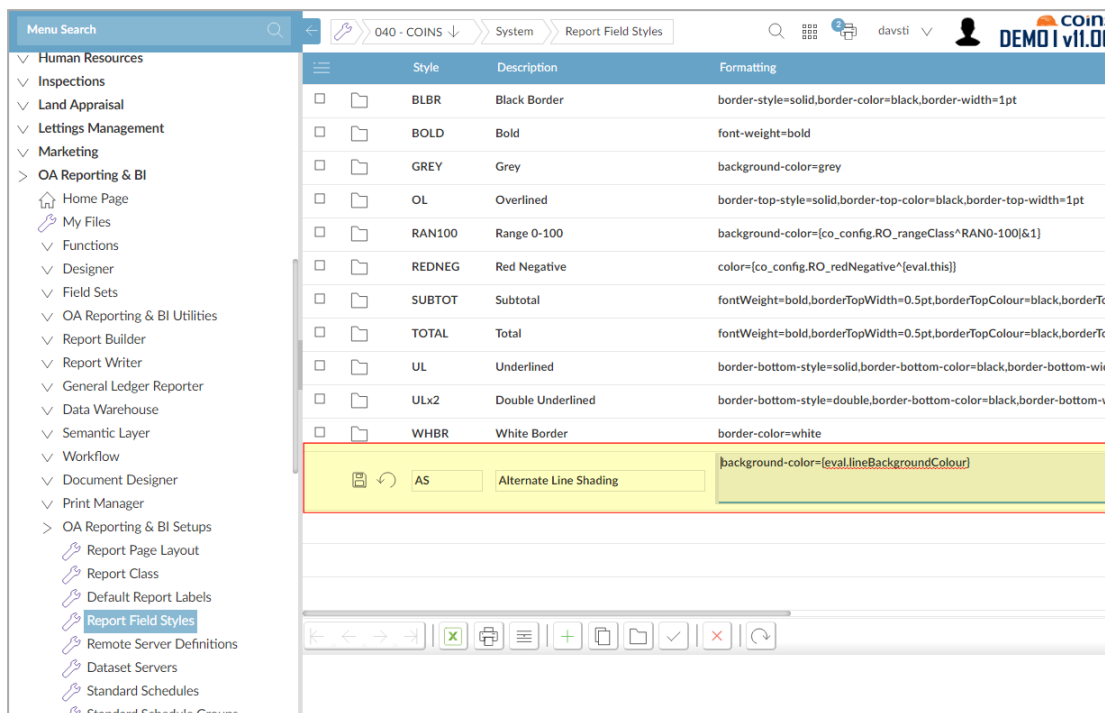
Field styling is not cumulative and as such a unique field style will need to be created for each existing field style where an alternating background colour is required.

The following steps will be performed via the COINS OA web interface.

Log into the COINS OA web interface and navigate to OA Reporting & BI → OA Reporting & BI Setups → Report Field Styles as shown below:

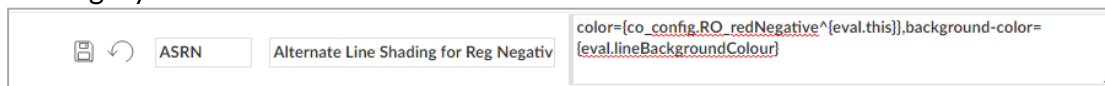


1. Add a new report field style that references a variable, which will hold the required line shading colour:



The Style field must be a unique alphanumeric;  
The Formatting field must contain the name of a variable that will be created in the report or form later in the How-To guide. In the above example, the variable is named: lineBackgroundColour

2. Add all further new report field styles where the alternate line shading will be required. Multiple formatting options can be applied by separating each formatting option with a comma (,) character. For example, where the existing “Red Negative” report field style is used where line shading will be required, then create a new report field style for the combination of the “red Negative” style and the alternate shading style:



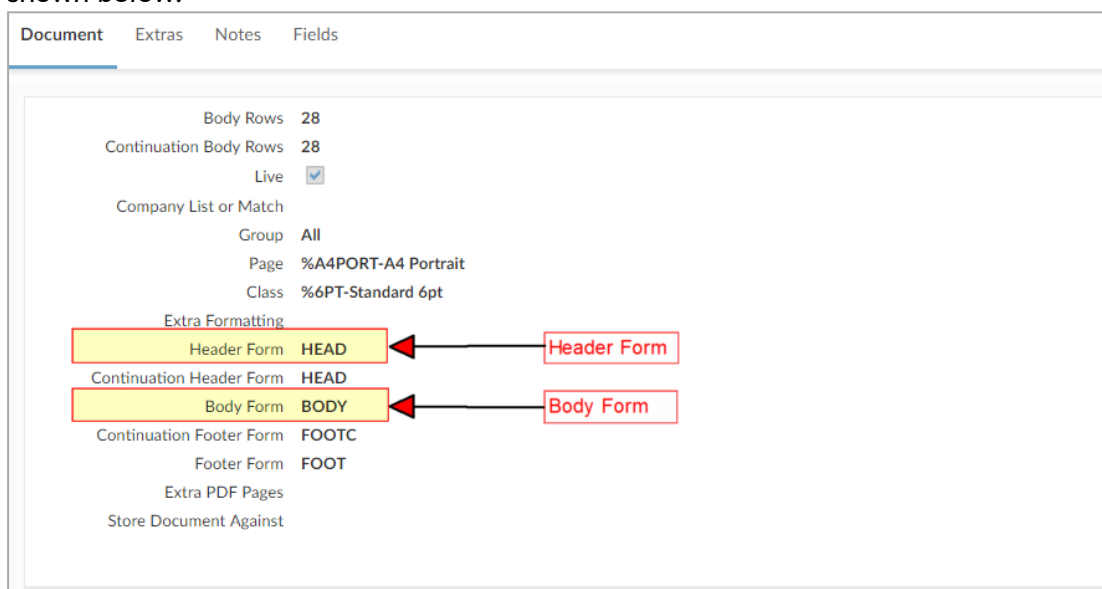
Once all required report field styles has been created, then proceed to the next task.

## 7.10.4 Applying to forms

Applying alternate line shading to forms is a little different to how it is applied to reports. Forms do not have defined sections as reports do. Instead “forms” can be defined on the fly and “forms” do not have any properties of their own.

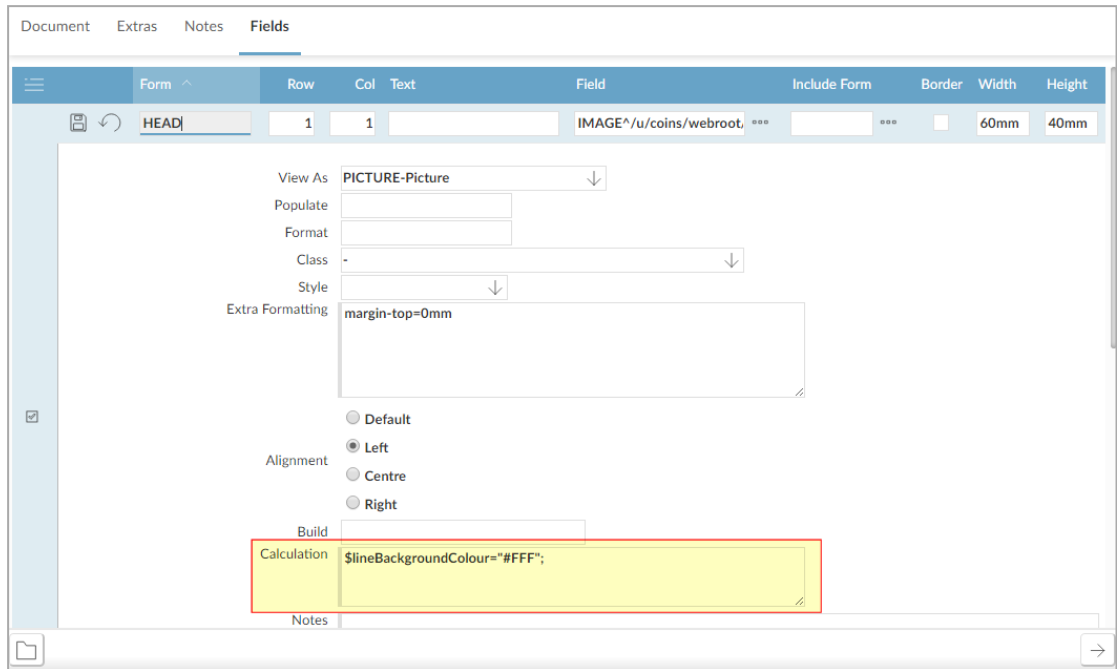
The following steps will be completed using the COINS OA web interface.

1. Log into your coins environment using the COINS OA web interface. Navigate to the OA Reporting & BI → Document Designer → Document Designer menu option and drill into the form that you want to modify. Go to the Document tab as shown below:

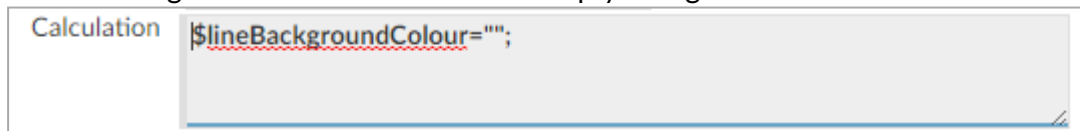


Note the names of the Header Form and the Body Form

2. We need to initialise the variable that has been defined in the alternate line shading report field styles, created in Create report field styles. To do this go to the Fields tab and open the first field that is part of the named Header Form and add the calculation as shown below:



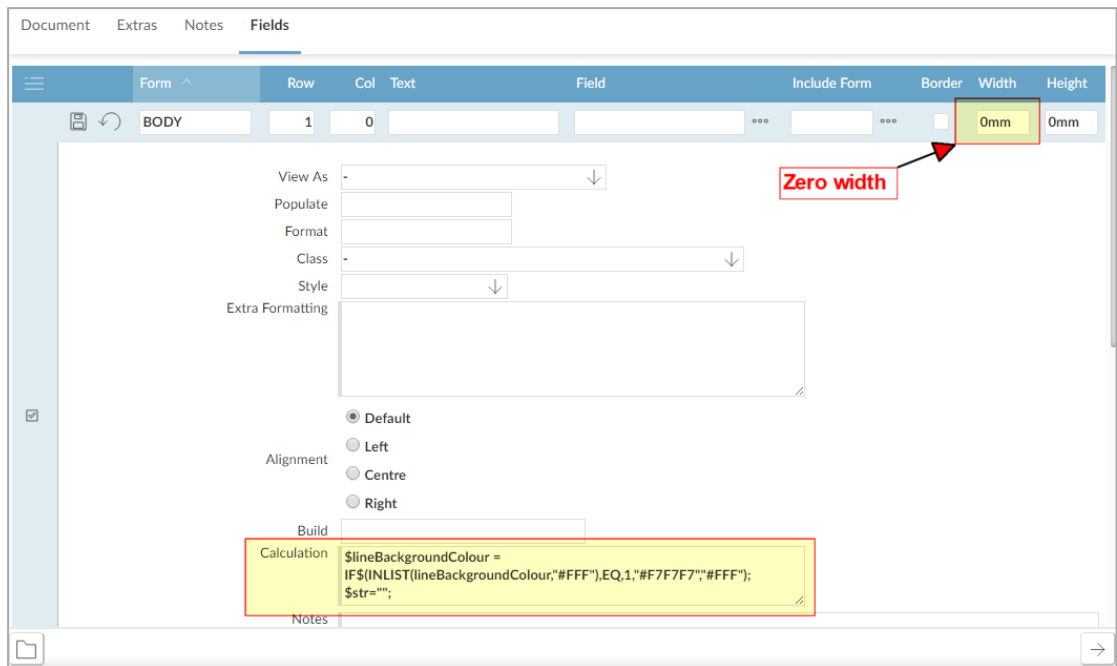
This will initialise the lineBackgroundColour variable to white and the first line of detail will be shaded. If your preference is to have the first line unshaded, then set the lineBackgroundColour variable to an empty string:



Save your changes.

- Identify the first field that is associated with the named Body Form and insert a new field on the same Row number but with a lower Column number so that it is ordered before all other Body Form fields, but does not create an additional row in the Body Form. You may need to reorder the existing Body Form fields to be able to insert this new field. The new field should be defined as follows:





Note the zero width field and the calculation:

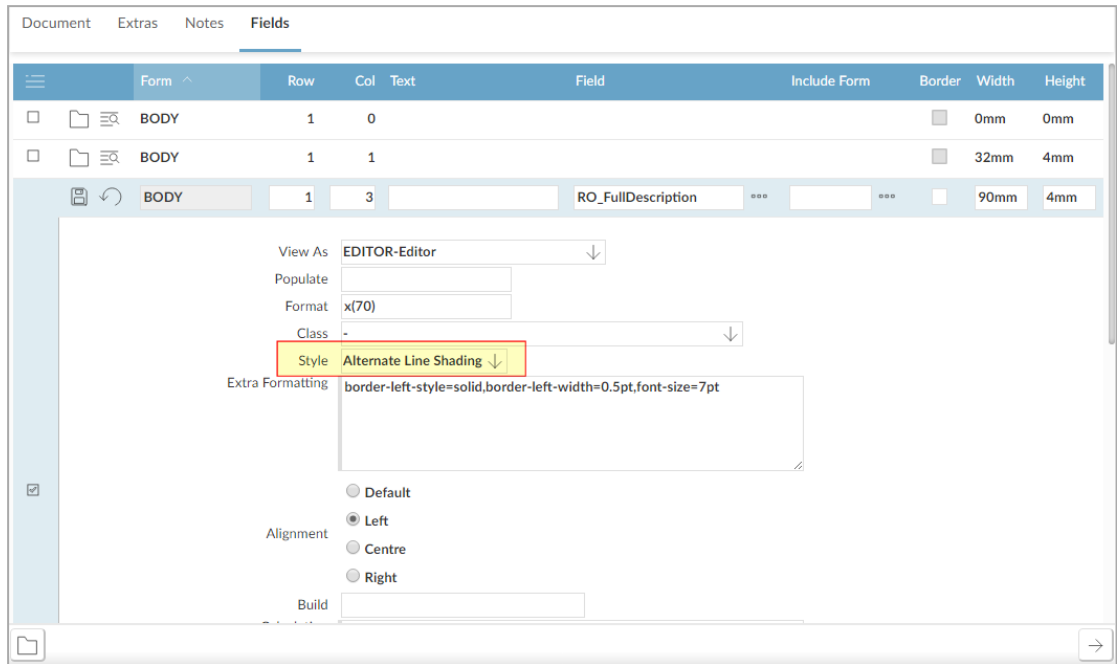
```
Calculation $lineBackgroundColour =
IF$(INLIST(lineBackgroundColour,"#FFF"),EQ,1,"#F7F7F7";"#FFF");
$str="";
```

Each line of the Body Form will alternate between #FFF (HTML colour code for white) and #F7F7F7 (HTML colour code for an off-white shade). These colour codes can be tailored to your specific requirements.



Note that some COINS forms, for example Purchase Orders, contain additional forms that may appear in the place of the named Body Form. In these instances, step 3 will need to be repeated for each of these forms if they are to also be alternately shaded.

4. Now visit all other fields associated with the named Body Form and set the Style field to the appropriate report field style:



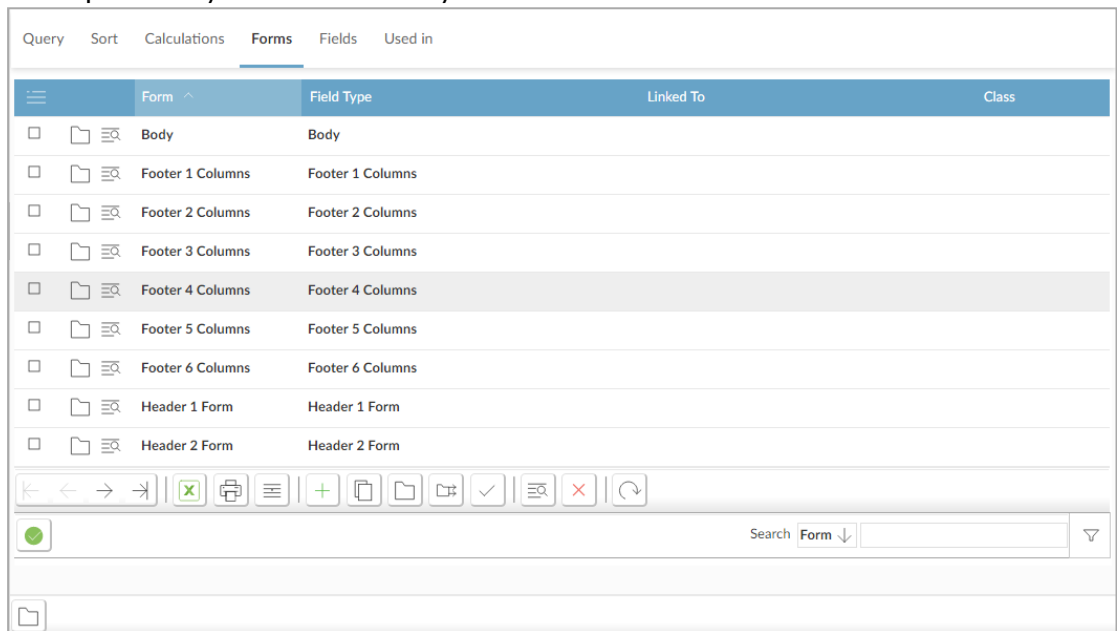
Once all fields of the named Body Form have been modified to use the created report field style, test the changes to ensure that the form lines are shading as expected.

## 7.10.5 Applying to reports

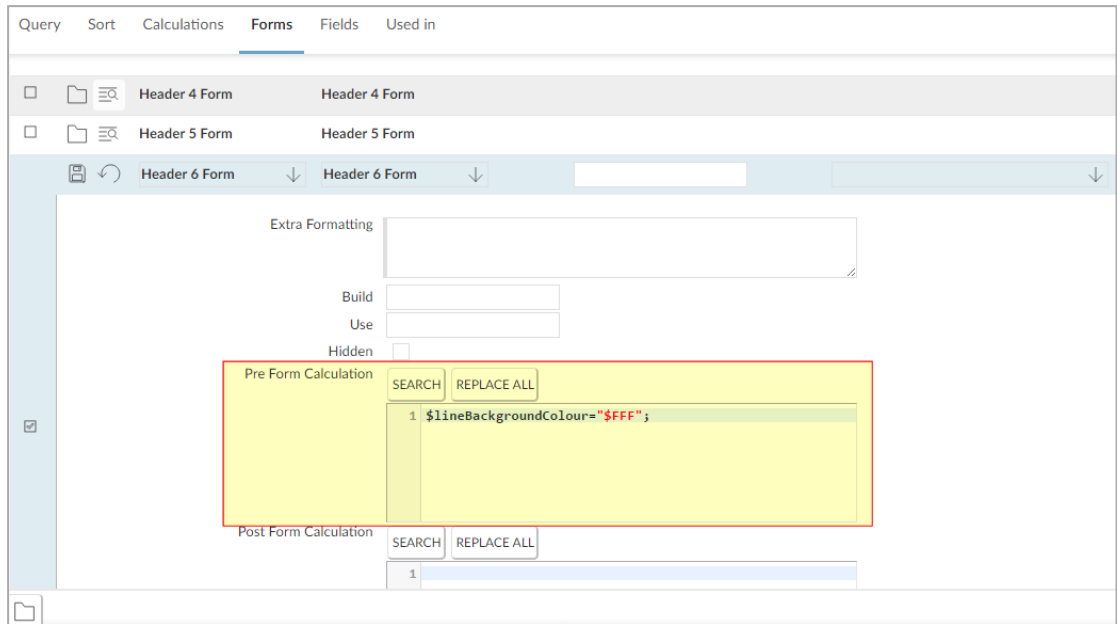
To apply alternate line shading to COINS reports, we will use the calculation fields on the report forms rather than in “hidden” fields, as was necessary in forms.

The following steps will be completed using the COINS OA web interface.

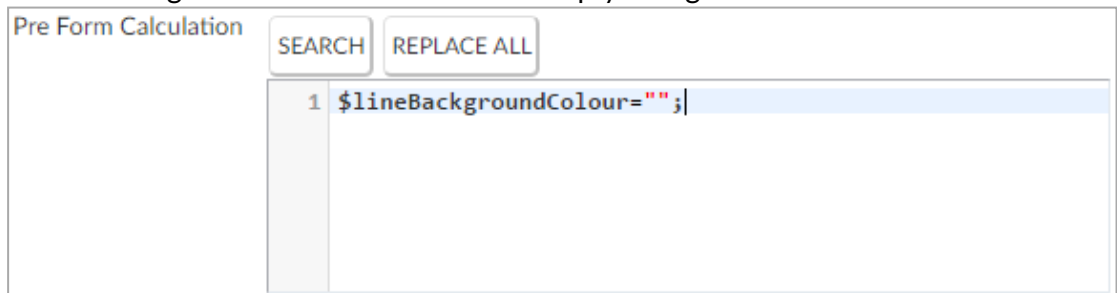
1. Log into your coins environment using the COINS OA web interface. Navigate to the OA Reporting & BI → Designer → Report Designer menu option and drill into the report that you want to modify. Go to the Forms tab as shown below:



2. Identify the inner most Header Form. We will use this form to initialise the line shading variable defined in Create report field styles. In this example we have defined the variable: lineBackgroundColour. Open the inner most header form and add the calculation as shown:

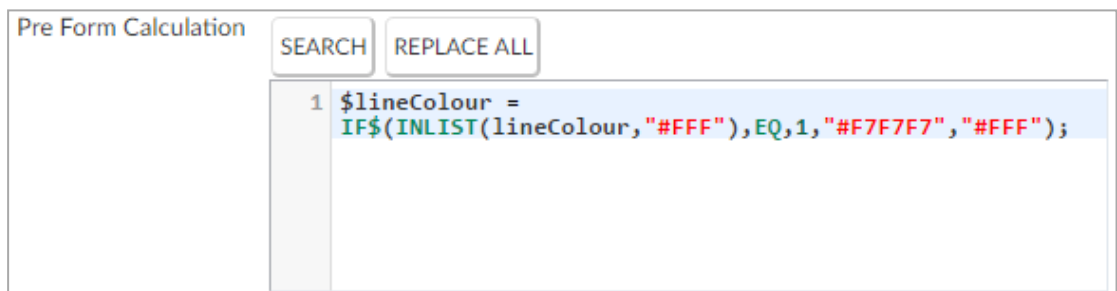
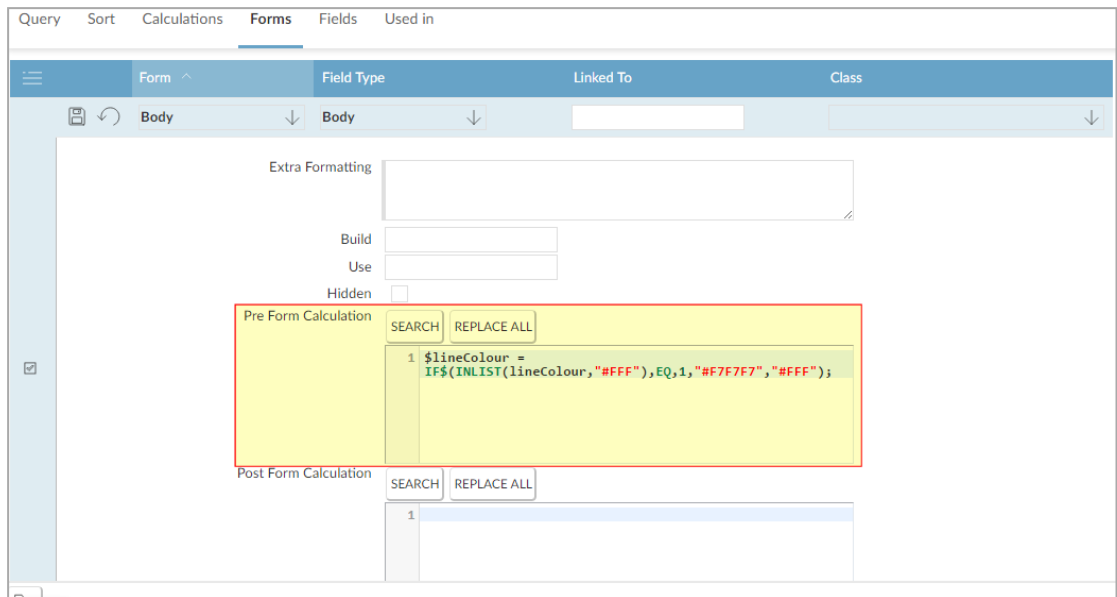


This will initialise the lineBackgroundColour variable to white and the first line of detail will be shaded. If your preference is to have the first line unshaded, then set the lineBackgroundColour variable to an empty string:



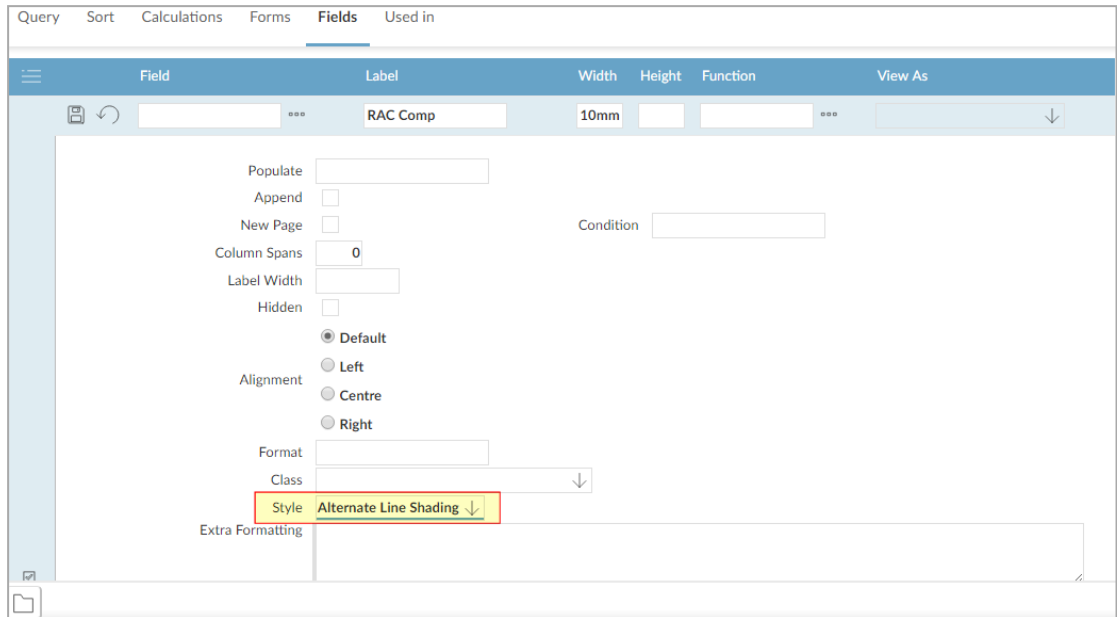
Save your changes.

3. Open the BODY form and add the calculation to alternate the line shading as shown below:



Each line of the Body Form will alternate between #FFF (HTML colour code for white) and #F7F7F7 (HTML colour code for an off-white shade). These colour codes can be tailored to your specific requirements.

4. Go to the Fields tab and visit all fields associated with the BODY Form and set the Style field to the appropriate report field style:



Once all fields of the BODY Form have been modified to use the created report field style, test the changes to ensure that the report lines are shading as expected.

## 7.11 Queries on Purchase Orders

It is recommended that you use `tip_type` in any query to ensure that it is indexed properly, the `tip_type` will either be MATERIAL (material, plant, asset, stock etc) or TRADE (subcontract).

The key thing to remember when reporting or enquiring on orders or order lines is that each time a variation is raised against an order a new `po_hdr` record, with associated `po_lines`, is created. This means you could end up with more than one line per order in your report or browse.

There are reporting fields (RO\_) available to assist with this and below is a standard query to get the latest order information (i.e. the latest committed variation).

```
FOR EACH po_hdr WHERE po_hdr.kco = {kco}
AND po_hdr.tip_type = "MATERIAL"
AND po_hdr.poh_mpo = "M"
AND po_hdr.poh_chgno = ""
{pohSelect},
```

### 7.11.1 po\_line v po\_item

As noted, a `po_line` record will be created for every line entered onto an order; commodity, text, clauses and will be created for every variation. However the actual detail of the lines are stored once each (only when the order or variation is committed) in the `po_item` table. This table will not contain text lines or clauses - only items with a value. There are extensive fields (both database and RO) on this table for reporting ordered, delivered, invoiced values and quantities.

To link to `po_item` from `po_hdr` (and then back to the latest `po_line` if required) append the following query to that above.

```
EACH po_item
WHERE po_item.kco = po_hdr.kco
AND po_item.tip_type = po_hdr.tip_type
AND po_item.poh_ordno = po_hdr.poh_ordno,
LAST po_line of po_item OUTER-JOIN
```

Another View of this query...

As the selection page will only work on the Original Order it may be the case that the above query is not what you are after. (ie the Selection on Order Date only refers to the Original Order). If this is not what is required then the following query may help. It will look at all orders which are committed and then look at all order lines which have a commodity code assigned and the query condition will remove the duplicate information and will report on the latest information on the order line.

```
Query FOR EACH po_hdr WHERE po_hdr.kco = {kco}
AND po_hdr.tip_type = material
AND po_hdr.poh_mpo = M
AND po_hdr.poh_committed = TRUE
{pohSelect},
EACH po_line of po_hdr WHERE po_line.pol_type = 'C'
```

Query Condition: RemoveVariedOrders



## 7.12 Calculations Overview

This section provides an overview of the calculation syntax used within the OA Toolset and provides some examples of its use.

Calculations can take place on any Database field and/or include any conditions required to produce a desired result. This section basically splits calculations down into three major areas.

- Calculations on Numerical Values
- Calculations on Character Values
- Calculations on Date Values

There are various areas in the BI toolset where calculations are utilised and the syntax used is consistent across all of these. Some of the areas where calculations can be used are as follows:-

### Report Writer

- Field Definition

### Report Designer

- Field Definition
- Initial Calculation
- Pre Report Calculation
- Post report Calculation
- Calculate Condition

### Page Designer

- Field Definition
- Initial Calculation

### DataSets

- Field Definition
- Initial Calculation
- Pre Report Calculation
- Post report Calculation

#### DataMarts

- Column Definition

#### Workflow

- Initialisation
- Stages

## 7.12.1 Mathematical Functions

+ - / *	Valid Operands are add, subtract, divide and multiply
^	Exponent (a to the power of b)
fact(n)	Returns factorial n (that is, $n*(n-1)*(n-2)*...$ )
pi	Returns the value of pi ( $\pi$ )

`jc_job.RO_ContractCosts^TD - jc_job.RO_ContractCosts^TP;`

It is necessary to ALWAYS surround any mathematical sign with spaces. An operand without spaces will cause an error. To remember this think of the minus sign which without spaces will be treated as a hyphenated word.

## 7.12.2 Calculation Field Rules

It is possible to include multiple calculations within the one calculation box - Each calculation is separated by a ; symbol (semi-colon). It is always the last calculation in the box which will appear on the report if the field is a calculated field.

If there is a 'Field' defined on the field record then it will be the 'Field' which is displayed although the calculations will execute.

Calculations can only be performed on data of the correct type, i.e. The Test in an IF statement can only be performed on an integer or decimal value.

Typical error messages which may be reported in the Log File are :

"Primary expected but found string".

"Invalid Character in Numeric Input ..."

The \$ sign at the front of a variable within the calculation box will indicate that the result of the calculation is a character value.

var- A simple variable will expect a numerical value.

\$var - defines the variable var as a string - it will therefore expect character format data and should be in quotes.

### 7.12.3 Variables

Values can be assigned to variables so that they can be used in a later calculation.

To assign a value to a variable use the format:

**Variablename=<value or calculation>**

```
jToDat='01/01/11';  
dProfit=dRevenue - dCosts;
```

'this' is a reserved word which will enable you to assign the current field value to a variable.

```
dValue=this;
```

DO use standard naming conventions, examples used by the BI Team and COINS Developers are :

- jDate - A prefix dates with j
- iCount - prefix integers with i
- dCosts - prefix decimal values with d
- cText - prefix text/character values with c
- lComp - prefix logical values with l

Using a convention like this should also prevent you hitting on the major Don'ts (using dictionary words as variable names).

DO always qualify fields when using them in the calculation fields within DataSets and Calculate Conditions. Eg. jc\_job.RO\_ContractCosts^TD

DON'T prefix any variable name with a numeral - it will fail.

DON'T use dictionary words as variable names, they may be commands in COINS or



PROGRESS, examples that will fail include Group, Type, Index, Count, Total - if you use prefixes as noted in the DOs above - you can use cGroup.

## 7.12.4 Debug

By turning debug mode on, all calculations will be verbose in the log file.

The log file is the last icon on the report status workbench -

The basic layout of the log file shows:

- > The variables currently being sent to the report from the Page (via Form Service Procedure
- > The Record Service Procedures being used within the report
- > If a DataSet is being used in the report the Temp Table will be defined and then the Query for the dataset.
- > The value of any replacements being made within the Report Query (Eg {kco} - You will see REPLACE: kco 1)
- > The Query used on the report and the value at the start of the query is the number of seconds the Query takes to run.
- > Report Selection - Indicates that the Report has completed.
- > FOP - Which is the PDF generate program does display some Warnings and Errors - Please follow the link for some common errors which cause the PDF to Fail

To turn debug on include debug(1);

This can be included in the initial calculation field and will be available to all calculations within a report unless debug(0) is entered to turn it off. Although putting a report LIVE with debug turned on will not cause an error there should be a process put in place to ensure that debug has been removed before a report is put 'LIVE' as debug can make log files unnecessarily large and can use disk space.

Example:



Report: NLEX13 [Lint](#)

Header Tables:

Body Table:

Notes:

---

Query Sort **Calculations**

Initiation Calculation:

1 debug(1);

Pre Report Calculation:

1

Post Report Calculation:

1

Results:

```

Start Time:03/03/2016 16:13:41.247+00:00
Database User :5
Report Process:9003
03/03/2016 16:13:41.264+00:00 Variables:1510
Parameters: &RS_DATE_1=&RS_DATE_2=&RS_rpt=NLEX13&RS_type=B&ExportType=P,X,S&mp$code=coinsstd&RS_copy=1&RS_eto=nigel.longley%40coins-global.com&RS_ecc=&RS_esubject=Home - NLEX13&RS_eattachName=
03/03/2016 16:13:41.265+00:00 Starting private sfdrsp.p
03/03/2016 16:13:41.269+00:00 Started private sfdrsp.p
03/03/2016 16:13:41.273+00:00 Running service: syfrep (syurep.p,pm99.p,pm99b.p)
03/03/2016 16:13:41.295+00:00 Starting private rfdrsp.p
03/03/2016 16:13:41.340+00:00 Started private rfdrsp.p
03/03/2016 16:13:41.344+00:00 Header Tables:
03/03/2016 16:13:41.347+00:00 Running service: ainrsp (returnTableRSP cos000.p,getTableRSP cos000.p,report syurep.p,syurep.p,pm99.p,pm99b.p)
ShowVariables:
1
03/03/2016 16:13:41.411+00:00 Setup Query Start
03/03/2016 16:13:41.412+00:00 Replace:kco 10
03/03/2016 16:13:41.426+00:00 Replace:kco 10
03/03/2016 16:13:41.468+00:00 Query Tables:ap_invoice
03/03/2016 16:13:41.475+00:00 Query: .008s for EACH ap_invoice WHERE ap_invoice.kco = 10 and ap_invoice.ain_balance NE '0' AND (TRUE) AND (TRUE) no-lock
by ap_invoice.avn_num
indexed-reposition
03/03/2016 16:13:41.570+00:00 Running service: avm-rsp (setBufferHandle ain-rsp.p,setRow syurep.p,report syurep.p,syurep.p,pm99.p,pm99b.p)
03/03/2016 16:13:41.622+00:00 Running service: avm-rsp (setBufferHandle avm-rsp.p,setRowid$ avm-rsp.p,setBufferHandle ain-rsp.p,setRow syurep.p,report syurep.p,syurep.p)
03/03/2016 16:13:41.666+00:00 Report Content Start
03/03/2016 16:13:41.666+00:00 ReportContent
03/03/2016 16:13:41.686+00:00 Report Content Prepared
dVAT=ap_invoice.ain_taxamt
03/03/2016 16:13:41.746+00:00 Running service: sys001 (getRowFieldValueFormat ain-rsp.p,getTableFieldValue cos000.p,expr coueval.p,expr coueval.p,evaluate$ coueval.p,repCalc syurep.p)
17.5
dVAT
17.5
$dVAT="17.5"
17.5
    
```

If only a specific calculation needs to be debugged, rather than display all calcs in the log it is possible to selectively switch debug on and off. Simply add debug(1) in front of a calculation in the calculation box of a page or report field and debug(0) after it

```
debug(1);$SectOrd_code=Substring$(co_extra.cex_cha_7,1,10);debug(0);
```



## 7.12.5 Curly Braces

The functionality of {}'s is to specify a place holder in fields and calculations into which a value can be passed.. When using {}'s around a field the use of quotes is required if the field is a character field. The use of double or single quotes is acceptable.

The only thing to be aware of is that when using '{field}' replacement on a character field is that if the information within the field could contain an apostrophe (for example- J O'Connor) then the apostrophe would cause close to the single quote and you will get a symbol not found(Connor) Error. To overcome this error the use of double quotes "{field}" is the answer.

The use of {}'s in calculations is possible on all field values **except within the DataSets and the calculate conditions on a report**. In these instances it is necessary to always qualify out the field with the table name.

```
{RO_ContractCosts^TD|0|{RS_glp_fdate__2}}
```

would be written as:

```
jc_job.RO_ContractCosts^TD|0|{RS_glp_fdate__2}.
```

The use of the table name is allowed in all calculations but whereas in most instances the formatting of the result is suppressed, within the calculate condition it is not and therefore the comma in a result of a figure in excess of 1,000 may result in an error in syntax in a calculation. (NB. Please note that the replacement on parameters of an RO field is still acceptable).

Within the OA reporter/screens we use curly braces {} as a method to pass values to a query or a report or a page. Enclosed within the curly braces you specify the commands, RS\_fields, or other data you need to communicate across or within objects. {kco} is a common usage, and is used to place the current logged in company number into the query.

The next example gets information from jc\_job and inherits the Company Number from the system, retrieving the company number the user is logged into.

```
FOR EACH jc_job WHERE jc_job.kco = {kco}
```

## 7.13 Datasets - Overview

A Data Set is a pre-defined Table which is created at the time of the query. Once the table is created then it can be accessed via either Report Writer or Report Designer. The benefits of using Data Sets are:

- Provide access to PROGRESS temporary table functionality
- Build single table of data from various COINS tables
- Build Data Sets for use in multiple reports
- Allow sorting by virtual fields
- Allow filtering by virtual fields
- Allow union of several Data Sets ( Useful for Cross Modular Reporting)
- Summarisation of data
- Simplification of data views for users

### 7.13.1 Creating a Data Set

Once the fields which are required have been identified the query to obtain those fields must be created. In this example the fields required have been identified as

- Contract Number –(job\_num)
- Contract Name – (job\_name)
- Contract Cost – (RO\_ContractCosts^TD)
- Contract Revenue – (RO\_ContractRevenue^TD)
- Contract Profit (RO\_ContractRevenue^TD – RO\_ContractCosts^TD)

To create a new Data Set go to the Data Set Definition option under the Designer Menu.

Click

Select a Data Set Name and Description for the Data Set in this example is has been named 'NLWJC\_PROF' with a description of 'Standard JC Profit/Loss DataMart'.

A Query must be assigned to the Data set, which in the following example is a simple query of each Contract which takes advantage of a 'Contract Selection' made at run time.

```
FOR EACH jc_job WHERE jc_job.kco = {kco}{jobSelect}
```

Create an identifier for the Data Set Table Name. In this example it is 'contract'.

Once created, the table name is always prefixed with a 'tt' when used, so in this case the table will be accessible in a report using a 'FOR EACH ttcontract' query.

Field	Description
Generate Program:	It is now possible to call COINS standard generate programs and include them into your Data Set. – Sample generates scr399,plr399,slr399,csr399. (Open item records).
Raw DB Query	This will bypass the Business Logic and get data directly from the Database so care should be taken on who is given access to Data Set

Field	Description
	Queries where this selection is selected. (ie: This will ignore all security).

There are three calculation fields which can be applied to the data set.

Field	Description
Initial calculation	This will calculate prior to the query executing thus allowing the variable to be used within the Query, e.g.  <code>\$sdate = date\$(datestring(co_config.glp_fdate^-12 {RS_glp_fdate}));</code>  Would allow a query to be created such as:  <code>FOR EACH jc_job WHERE jc_job.job_condate &gt;='{eval.sdate}'</code>
Pre calculation	This will take place after the query but prior to the fields calculating
Post calculation:	:This will take place prior to the data set closing

To move onto the Data Set fields, click

#### 7.13.1. Data Set Fields

When Adding fields to the table, you can give the field a Name that can either be same field name as the field in the source table or you can name the field something that will help Report Writer/Designer users identify the information in the field more easily. (Eg: location\_desc is assigned to the field jcl\_desc)

Each field should be given:

Field	Description
Label	This will become the name of the field when using the Data Set in queries/Reports etc. You may either using teh same name as the source field or assign a more user-friendly name. Spaces should not be used
Data Type	(eg Character, Date, Decimal, Integer, Logical)
Default Format	(NB. All formats are only defaults and can be overridden in reporting)
Source	The source can be any field (Database or RO) from the tables accessible via the query or it could be a calculation.

As an example, the sample of the fields which have been added to the Data Set are seen in the diagrams below.



## 7.13.2 Best Practice

- Only create fields in a DataSet that are going to be used in the equivalent report.
- It is better to 'Source' a field rather than to 'Calculate' a field for efficiency.
- It is recommended that any calculations that can be carried out on the report/page should be done at that stage.
- For performance reasons, {}'s are now replaced once at the start of the dataset (v10.22 onwards) so as to not replicate the process for every field in the dataset. This is not a problem in most instances as the {}'s used in RO fields are usually something along the lines of {RS\_glp\_fdate\_\_2} which is consistent across all records. It will mean however that you cannot put {}'s around field names; for instance '{job\_num}' is not possible because this replacement will differ on each record. It is imperative that fields like this are referenced with Tablename.Fieldname e.g. jc\_job.job\_num - It is always best practice to fully qualify fields in a calculation on a dataset as the field will not strip out formatting etc so the value 10,000 would cause a problem previously in an if statement as the comma would be read into the if statement causing the syntax to have too many parameters.
- To maximise efficiency, any field on a DataSet which is not at the lowest level of the DataSet query, should be taken out of the DataSet and initiated on the report. For example if you had a query on the DataSet which reads

```
FOR EACH jc_job WHERE jc_job.kco = {kco},  
EACH jc_costcode OF jc_job.
```

- Then fields from jc\_costcode should be included in the DataSet but any RO fields etc from the jc\_job table should be called on the report. To save unnecessary replication of calculations.
- It is good practice to Summarise at DataSet level, rather than at Report level if Summarisation is required. This is so that multiple records are not created and passed across to the report unnecessarily. This will reduce the load on the network traffic, especially where report servers are in use, and will reduce the amount of time the report takes to generate.

Debug(1); - Can be turned on to debug a dataset but should be removed when setting a DataSet live. Debug(1) should be defined as a calculation in the Initial Calculation box. Once debug is turned on then all calculated fields will be verbose in the Report Log File.

Data Sets can be exported from one environment to another by simply choosing the Export Data Set Definitions option from the Designer menu.



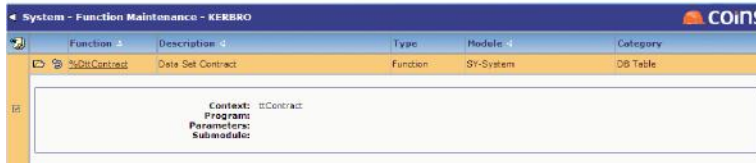
Enter the Data Set Name or use the lookup facility to find the relevant Data Sets and click Next

Once the Data Set has been exported to the Definition Data Window use the standard windows select all(Ctrl-A) and copy(Ctrl-C) functions to copy the data from the Definition Data Window and then Paste(Ctrl-V) to the Import Data Sets Definitions window also found Menu below in the environment you are wanting to add the Data Set after which you should then click the save icon.

A Data Set cannot pre-exist so if there is amendment required to a Data Set it must first be removed prior to importing

### 7.13.3 Granting Access to Data Sets for Other Users.

Once the Data Set setup has been completed you can allow access to the relevant users by creating a %DttContract Function via Function Maintenance.



Access to this function is then granted via the standard Function Security procedure.

The Data Set can be enabled to be accessed by Report Writer by the creation of a Report Writer query accessing the ttContract table.



### 7.13.4 Rules for Keys

It is possible to summarise DataSets by adding a 'KEY'. If the query is looking at all Contract (jc\_job) records but the requirement in the report is to be by Contract Location then a 'KEY' can be set at kco/jcl\_loc (ie: Tick the Key box on both of these fields). In this instance only one record will be created on a unique find of Company/Contract Location. Any numerical fields are accumulated whilst character fields are assigned where there is common data where records share the same 'Key' details. (If it finds character fields which differ then the value of the field will be blank).

If there is a requirement to do a calculation once all of the records for the 'Key' are accumulated then you can tick the 'Recalculate Summary'. (Eg: An example of where this might be required is when a percentage is required).

Note: Any field which is used as a key CANNOT have a value of blank.

### 7.13.5 Cross Modular Reporting

To achieve Cross Modular Reporting two independent Data Sets can be created using different Source information but containing the following commonalities

- Same Table Name
- Same Field Names (in same order)
- Fields must have the same DataType (ie. Char,Int,Dec,Logical,Date)

The datasets can then be both called on the report and the common table name will unionise the data. An example of such an application might be a Purchase Ledger and SubContract Ledger Open Invoice Dataset.

### 7.13.4 Using Data Sets in the Query Editor

The Query Editor allows fast access to information in the COINS Database via the Business Logic. It also allows export to Microsoft Excel.

It is possible to reference a Data Set from the query editor. Simply referencing the data set and running will return the complete contents of the Data Set - No query is required.

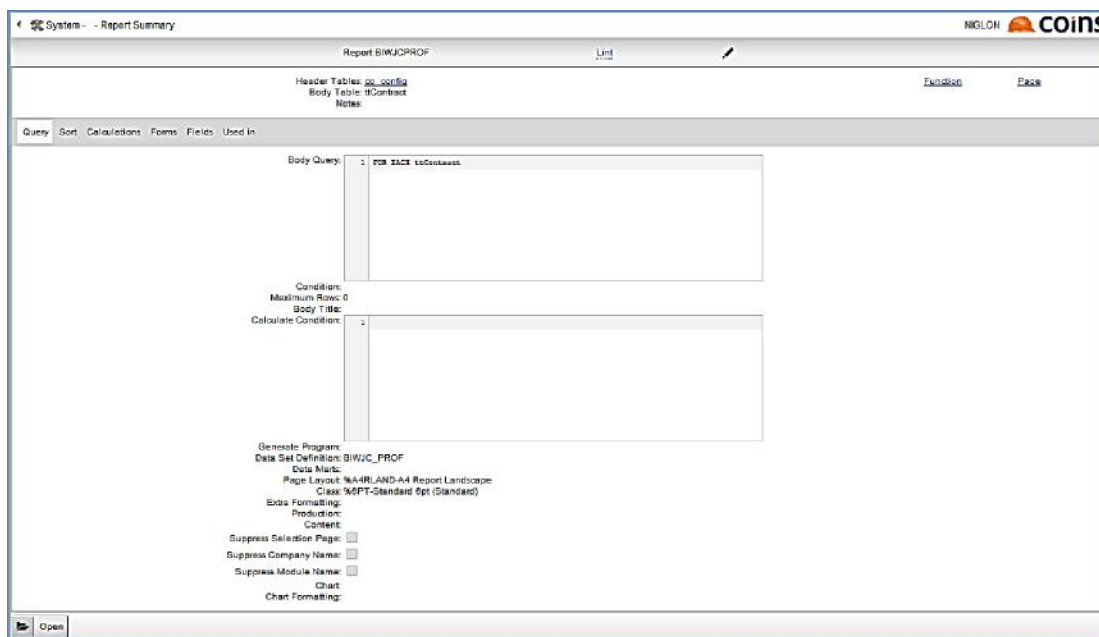
If the Dataset query takes a parameter replacement field (i.e. {RS\_year}) then you can set the parameters by adding them to the URL on the Query Editor Page. (i.e. &RS\_year=2009)

NOTE : the full Data Set will be generated before any results are returned so be warned - a large data set may be slow.

This has allowed the extract to Excel to be more flexible as limits in the information being passed limited both the query length and number of fields that could be accessed through the editor. Referencing a dataset code greatly enhances the ability to extract data this way.

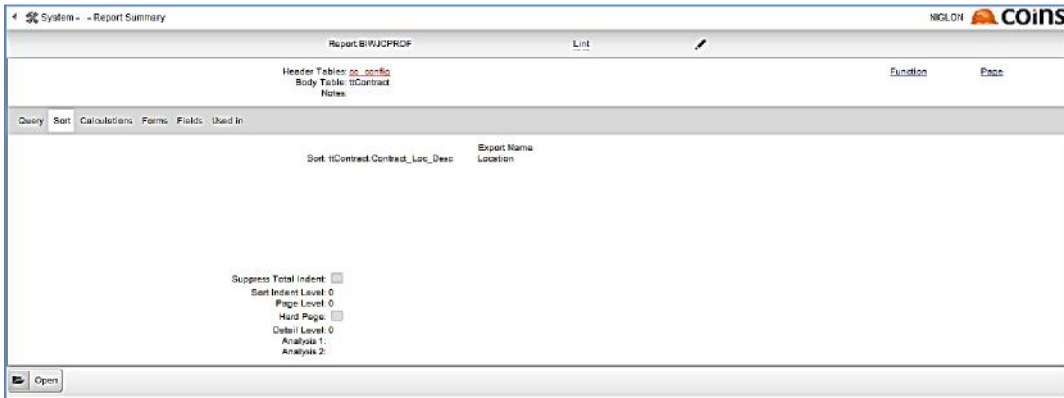
### 7.13.7 Using the Data Set in Queries

The Data Set can now be used in Queries in Report Designer. Using our example dataset, the body table should be ttContract, the body query should be - FOR EACH ttContract and the Data Set Definition would be BIWJC\_PROF.

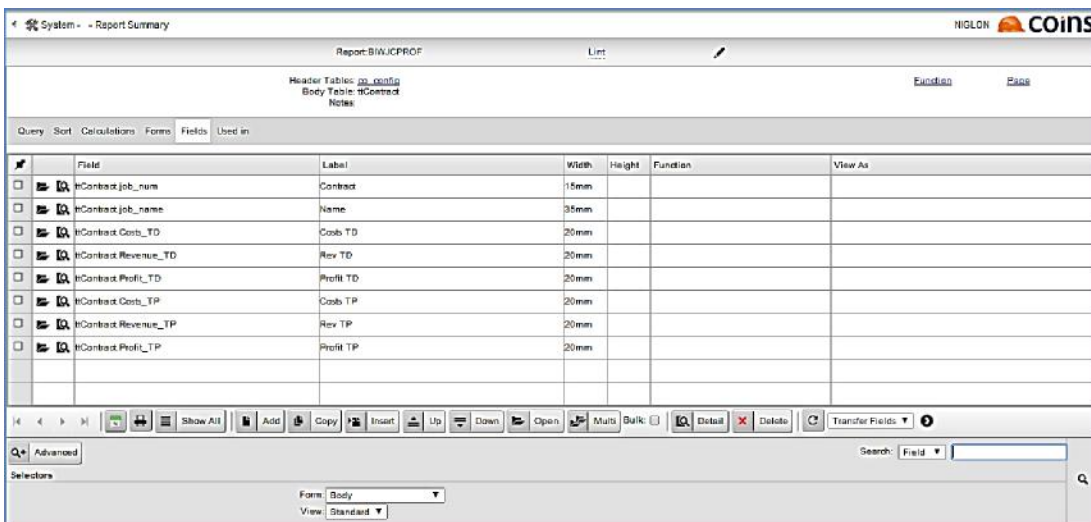


Note that you can use more than one dataset in a query. To do this enter each dataset name in the Data Set Definition field separated by a comma. You will need to reference each ttTablename as appropriate in the body query.

It is possible to sort on any field from within the Data Set – In the example below we are using a field which has been populated with the virtual field jcl\_desc with is the Location Description. This will put the contract in alphabetical order of the locations to which they belong. (NB. It is always necessary to qualify the field name with the Data Set table name in the sort)

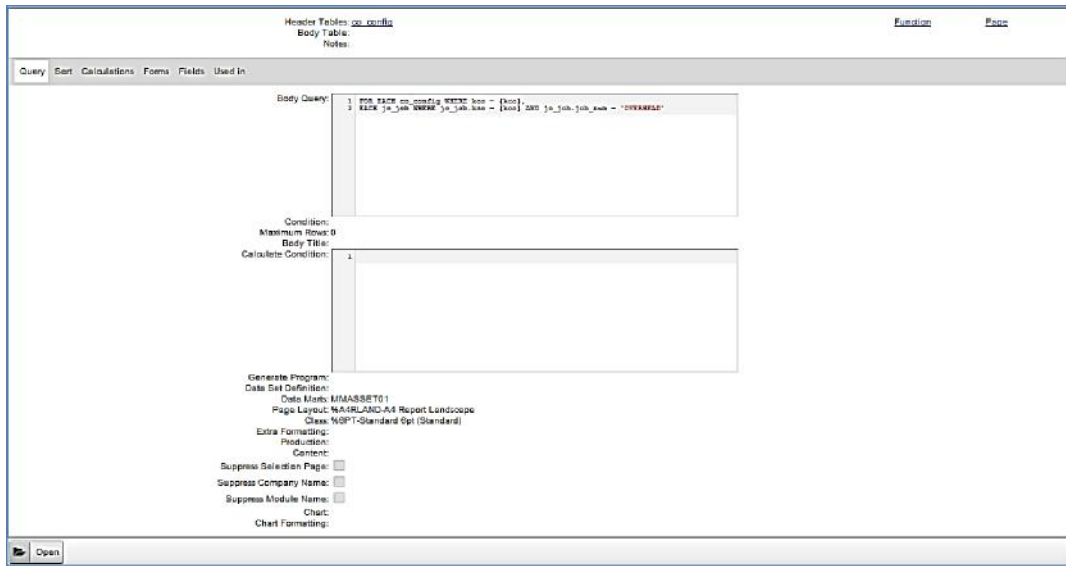


Fields can then be added to the report in the standard way. It is always good practice to qualify the field names also with the Data Set table name although it is not mandatory.

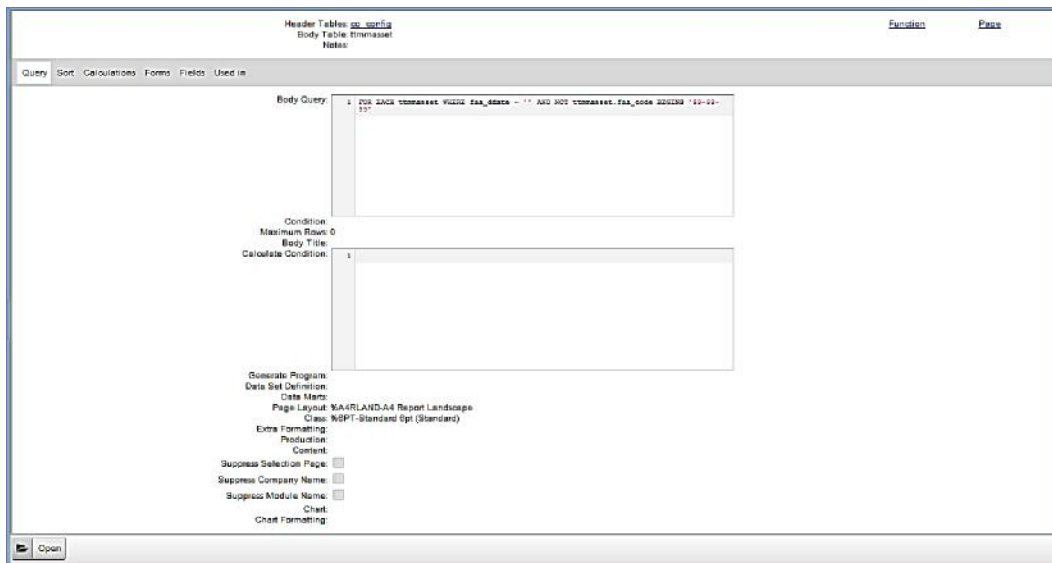


### 7.13.7.1n-Line Reports

For in-line reports, specify the dataset(s) to be used on the container report e.g.:



Then for each inline report, only use the ttTableName in the body query of each report but do not specify the dataset e.g.



This will ensure that the datasets are only built once and then data then shared across each report that uses it. If you specify the dataset name in the in-line reports, the dataset has to be rebuilt for that report which will have an impact on performance.

### 7.13.8 Report Pre-Processing (Syuds)

Reports in OA have mainly been banded or grouped row reports. There are a few hard coded exceptions where a grid or matrix of data is presented over a number of pages of the report.

Pre-Processing provides a generic mechanism to pivot a set of data in to columns and if required to page over multiple pages to allow any dataset to be processed in to further datasets that are suitable for printing in OA as matrix of data.

Further pre-processing methods are also provided to manipulate the data set (temp tables), however they are produced, and create further temp tables with the processed data. Methods are provided for Union, Merge, and Sum. There is also a debug option provided to dump the contents of a dataset to the log file and a Store option to store a dataset in a datamart extract.

### 7.13.8. Syuds.Calc

syuds.calc allows you to create a new table with calculated fields in it from other fields in the query.

You use calcTable to set the input query (multiple table) and output table name (e.g. Calc to produce a table called ttCalc)

You then run calc for each field you want to add to ttCalc and pass the following parameters:

name,

data type,

extent,

label,

format,

calc string

**Calcexec** then builds the ttCalc table, adds the fields you defined and runs the query and calculates for each row.

Can be very useful for adding extra stuff alongside existing records

The ttCalc record is created with rowid fields for each of the tables in the query so you can then join to it in a report query

syuds.calc example

Dataset Query:

```
FOR EACH jc_job WHERE kco = {kco}
```

Dataset Post Calculation:

```
method('syuds.calcTable','FOR EACH ttcontract','calc');  
method('syuds.calc','lastYTD','decimal',0,'Last YTD','','ttcontract.job_costsTD - ttcontract.job_ costsTY');  
method('syuds.calc','lastPTD','decimal',0,'Last PTD','','ttcontract.job_costsTD - ttcontract.job_ costsTP');  
method('syuds.calcexec');
```





Dataset Fields:

**Figure 26: Output: ttContract**

**Figure 27: Output: ttCalc**

#### 7.13.8. syuds.Debug

This method causes the contents of a dataset to be exported to the log file. Useful for seeing the results at various stages of pre-processing.

The parameters are table name, condition, fields (defaults to all fields in the dataset) and the number of records (if zero specified then 10 will be output).

#### **Figure 28: Dataset Post Calculation**

In this example, the first 10 contents of dataset table ttContract will be output

```
Method('syuds.debug','Contract','',0);
```

Output of the data in the log file is in CSV format suitable for pasting in to EXCEL.

Figure 29: Example output:

```

① 195.40.14.50/cgi-bin/oatrainbi/JC001295.log?kco=10&pvCIlevel=1&pvCISibling=10&TopMenu=%25WHOME&pvFrame=F%2C1583%2C620
PM01
costs_td = jc_job.RO_ContractCosts^TD
1787
rev_td = jc_job.RO_ContractRevenue^TD
0
prof_td = rev_td - costs_td
-1787
costs_td = jc_job.RO_ContractCosts^TP|0|
0
03/11/2016 11:27:57.266+00:00 PostRunQuery
03/11/2016 11:27:57.266+00:00 PostDataSet
Method('syuds.debug','Contract','','0')
DEBUG=niglon METHOD SIGNATURE=PROCEDURE,,INPUT pcTable CHARACTER,INPUT pcCondition CHARACTER,INPUT pcFields CHARACTER,INPUT piCount INTEGER
DEBUG=niglon SET-PARAMETER 1 INPUT CHARACTER Contract
DEBUG=niglon SET-PARAMETER 2 INPUT CHARACTER
DEBUG=niglon SET-PARAMETER 3 INPUT CHARACTER
DEBUG=niglon SET-PARAMETER 4 INPUT INTEGER 0
03/11/2016 11:27:57.267+00:00 Debug: Contract 0
03/11/2016 11:27:57.268+00:00 Query: FOR EACH ttContract
kco,job_num,job_name,costs_td,rev_td,prof_td,costs_td2
"10","10000","Penny Hill Estates","12550","156300","143750","12000"
"10","100010","Purchase Land","0","0","0","0"
"10","1001","PF Training Project","0","0","0","0"
"10","10010","New Wing","1200","0","-1200","0"
"10","1005","Baxter Building Refurbishment","0","0","0","0"
"10","1007","Hamptons Hospital Construction","57600","0","-57600","0"
"10","11000","","0","0","0","0"
"10","11001","","0","0","0","0"
"10","11002","","0","0","0","0"
"10","1111","Milton Keynes Building","4300","0","-4300","0"
03/11/2016 11:27:57.268+00:00 Dataset Complete: NLM2DS1
03/11/2016 11:27:57.269+00:00 Generate End
03/11/2016 11:27:57.269+00:00 Setup Query Start
03/11/2016 11:27:57.271+00:00 Query Tables:ttcontract
03/11/2016 11:27:57.271+00:00 Query: .001s for EACH ttcontract no-lock
by ttcontract.job_num
indexed-reposition
03/11/2016 11:27:57.272+00:00 Report Content Start
03/11/2016 11:27:57.272+00:00 ReportContent
03/11/2016 11:27:57.277+00:00 Report Content Prepared
03/11/2016 11:27:57.549+00:00 Rows Processed: 75
03/11/2016 11:27:57.550+00:00 ReportSelection
03/11/2016 11:27:57.586+00:00 Generate PDF Start
Nov 3, 2016 11:27:58 AM org.apache.fop.layoutmgr.table.TableLayoutManager getNextKnuthElements
INFO: table-layout="fixed" and width="auto", but auto-layout not supported => assuming width="100%"
Nov 3, 2016 11:27:59 AM org.apache.fop.layoutmgr.table.TableLayoutManager.getNextKnuthElements
INFO: table-layout="fixed" and width="auto", but auto-layout not supported => assuming width="100%"
03/11/2016 11:27:59.708+00:00 Generate PDF End
03/11/2016 11:27:59.715+00:00 Report Content End
03/11/2016 11:28:00.145+00:00 Email accepted for delivery. Srv smtpuk.coins.local. Port 25; To: nigel.longley@coins-global.com
03/11/2016 11:28:00.147+00:00 Delete FSP: jcfrep.p
End Time:03/11/2016 11:28:00.147+00:00
Timings: Startup- 29 Generate- 839 Prepare- 3 Production- 314 PDF- 2122 Completion- 439 Total- 3746
    
```



#### 7.13.8. \$yuds.Delete

This method deletes a dataset/temp tables from the report. It should be used if the dataset that has been generated is no longer required but might be rebuilt or reused later in the report (typically on inline reports).

```
Method('syuds.delete','CostRev,Budget');
```

The parameter is table names. The example would delete ttCostRev and ttBudget.

#### 7.13.8.4 syuds.Filter

This method causes the contents of a dataset to be copied to an identical dataset except that the records in the output dataset are filter based on a condition passed to the method.

```
Method('syuds.filter','CostRev','tdate="31/01/13"', 'CostRevJan');
```

The parameters are table name, condition, output table name.

The above example would take records from ttCostRev and filter on a condition clause where field tdate was equal to 31/01/13. The output records would be in ttCostRevJan which would have the same fields as ttCostRev.

This method is useful on inline reports to filter out a set of records from the containing report for processing or display on the inline. See also delete() method to delete this dataset once it has been used.

Also useful to filter a set of records in to a new temp table before storing using store().

#### 7.13.8.5 syuds.GroupQuery

**GroupQuery** came about because of report builder and is similar to sum except that they do multiple sums at different levels.

For example:

group by kco by jgr\_group by job\_num would create a summary record for:

level 0 (grand total),  
level 1 (kco),  
Level 2 (group),  
Level 3 (job\_num).

It equates the footer forms in the report builder and the report footer (level 0).

**GroupQuery** allows multi table query and takes the form:

Method('syuds.GroupQuery','[query string]','[output table]','[keys]','[sum fields]')

**Group** is just a simpler form of GroupQuery (for a single table).

**Group** takes the form:

Method('syuds.Group','[Source Table]','[Condition]','[output table]','[keys]','[sum fields]')

It actually then runs groupQuery with "FOR EACH XXX WHERE condition"

Example:

```
Method('syuds.group','Source','drev>1000','pcTable','pcKeys','SumFields');
```

Leaving fields to sum blank will do all decimals

Example:

Dataset Query:

```
FOR EACH jc_job WHERE kco = {kco} AND CAN-DO('{RS_job_num_3}',job_num)
```

Dataset Post Calculation:


```
method('syuds.calcTable','FOR EACH ttContract','Calc');
method('syuds.calc','lastYTD','decimal',0,'Last YTD','','ttContract.job_costsTD - ttContract.job_costsTY');
method('syuds.calc','lastPTD','decimal',0,'Last YTD','','ttContract.job_costsTD - ttContract.job_costsTP');
method('syuds.calccexc');
method('syuds.sum','Contract','','Location','jcl_loc','jcl_desc','');

method('syuds.group','Contract','','Group1','jcl_loc','job_costsTD');
method('syuds.groupquery','FOR EACH ttContract, EACH ttCalc WHERE ttCalc.ttContractRowid=ROWID(ttContract)','Group2','jcl_loc','');
```

Dataset Fields:

Field	Key	Extent	Label	Data Type	Format	Source
job_num			Contract	Character	x(8)	jc_job.job_num
job_name			Name	Character	x(30)	jc_job.job_name
jcl_loc			Location	Character	x(8)	jc_job.jcl_loc
jcl_desc			Location Description	Character	x(30)	jc_job.jcl_desc
jgr_group			Group	Character	x(8)	jc_job.jgr_group
jgr_desc			Group Description	Character	x(30)	jc_job.jgr_desc
job_costsTD			Costs TD	Decimal	->>>.>>>.>>9.99	jc_job.RO_contractcosts*TD
job_costsTY			Costs TY	Decimal	->>>.>>>.>>9.99	jc_job.RO_contractcosts*TY
job_costsTP			Costs TP	Decimal	->>>.>>>.>>9.99	jc_job.RO_contractcosts*TP

Output:

20 - Housebuilders QA - System - Query Editor Nigel Longley 

Stored:  Load Save Delete

Query: 

Search Replace all  
 1 FOR EACH ttGroup1

Fields:

Data Set:

Condition:

Timeout(Seconds):

Maximum Rows:

Extra Parameters: 

&RS\_job\_num\_3=1000

Run Export Create Dataset

level	kjcl_loc	count	job_costsTD	job_costsTD_max	job_costsTD_min	job_costsTD_avg
0		1	-71124.4	-71124.4	-71124.4	-71124.4
1	WARKS	1	-71124.4	-71124.4	-71124.4	-71124.4



20 - Housebuilders QA - System - Query Editor Nigel Longley

Stored: syudscalcalc Syuds Calc Example

Search

Query: 1 FOR EACH ttGroup2

Fields:

Data Set: Timjob

Condition:

Timeout(Seconds): 10

Maximum Rows: 100

Extra Parameters: &RS\_job\_num\_3=1000

level	kjcl_loc	count	job_costsTD	job_costsTD_max	job_costsTD_min	job_costsTD_avg	job_costsTY	job_costsTY_max	job_costsTY_min	job_costsTY_avg	job_costsTP	job_costsTP_max	job_costsTP_min	ip
0		1	-71124.4	-71124.4	-71124.4	-71124.4	4273	4273	4273	4273	200	200	200	20
1	WARKS	1	-71124.4	-71124.4	-71124.4	-71124.4	4273	4273	4273	4273	200	200	200	20



### 7.13.8. syuds.Merge

This method will combine selected columns from two or more dataset (with a common key) in to a new dataset.

Suppose we have a dataset ttCost

Kco	Job_num	dCosts
100	1001	100
100	1002	150

And a dataset ttRev

Kco	Job_num	dRev
100	1001	200
100	1002	250

Then the resulting dataset (ttCostRev) might be

Kco	Job_num	dCosts	dRev
100	1001	100	200
100	1002	150	250

This is achieved in a report initialisation calculation as follows:

```
Method('syuds.mergeKeys','kco,job_num');  
Method('syuds.mergeTable','cost',' ',' ');  
Method('syuds.mergeTable','rev',' ',' ');  
Method('syuds.mergeExec','CostRev');
```

mergeKeys is used to specify the unique keys used to merge the data.

mergeTable is called once for each table to be merged. The first parameter is the dataset name (without the tt prefix), the second parameter is the condition to be applied to this set of records, the third parameter is the key field names in this table (defaults to the mergeKeys),

the fourth parameter is the field names (in this table) to be combined, the fifth parameter is the name of the fields in the output dataset.

e.g. `Method('syuds.mergeTable','cost','WHERE dCosts > 0','kco,job_num','dCosts','dMyCosts');`

would take only records with costs greater than zero and using kco and job\_num from this record write dCosts in to a combined field called dMyCosts.

A short version is also available with default options.

`Method('syuds.merge','cost,rev','kco,job_num','CostRev');`

This is equivalent to the series of calls above. Datasets ttCost and ttRev are combined using kco and job\_num (in all tables) and all fields from the two datasets are combined and returned in ttCostRev.

Decimal values will be summed in the combined dataset. All other field types will be written from the last record to be combined. It is expected that the record to be merged would already be unique on the keys.

### 7.13.8. Syuds.Pivot

A new program has been introduced that is designed to be called in the initialize calculation of an OA report. It is handed a dataset name and a series of criteria and transforms the named dataset in to three dynamic datasets which are designed to be easier to print in a matrix.

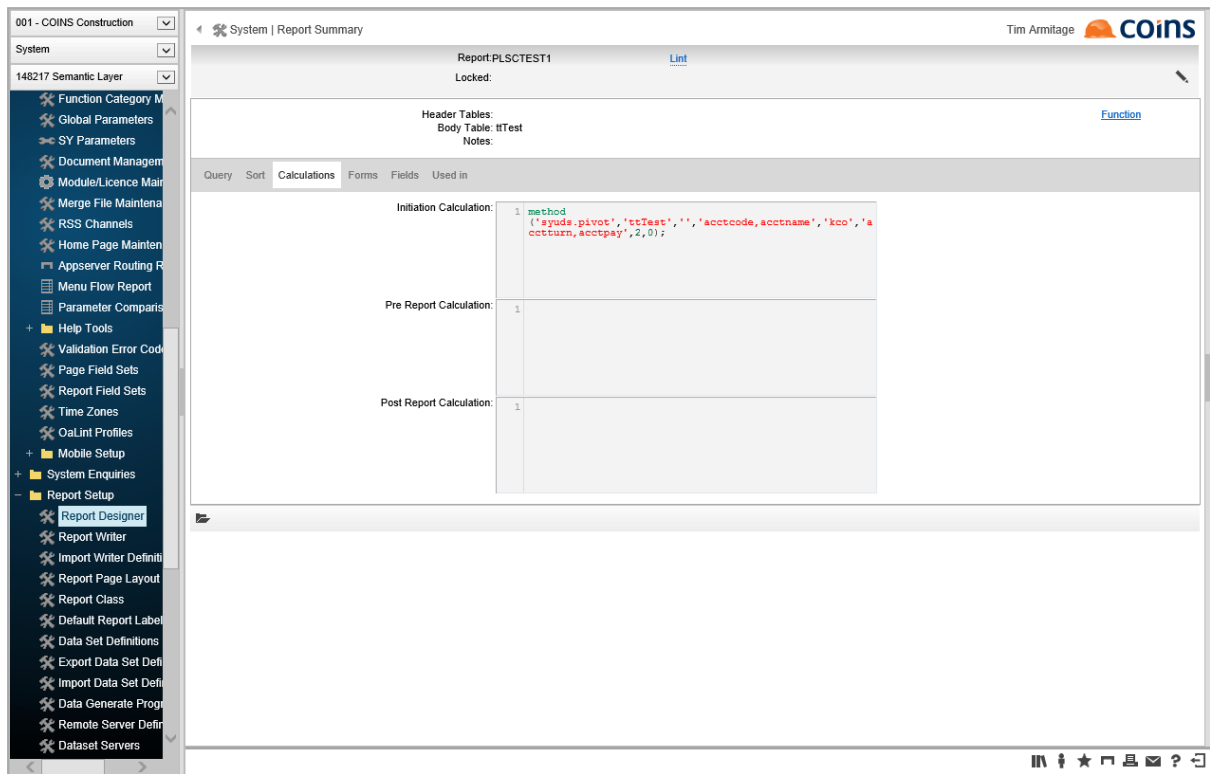
Syuds.Pivot take the following form:

```
method('syuds.pivot','[Source]','sumtype='[ ]','[Row Fields]','[Column Fields]','[Sum Fields]',
[Columns],[Total Column]);
```

Source	the name of the temp table source data that is required to be pivoted
SumType	sumtype="" is a condition to extract only selected records. SumType="" will extract all records
Row Fields	the fields that will be used to create unique rows in the resulting pivoted dataset
Column Fields	The field(s) which will be used to create the columns of the array of data in the resulting pivoted dataset
Sum Fields	the fields to be summed and added to the resulting pivot dataset
Columns	the number of columns per page
Total Column	1 means add a total column (the sum of all the columns on the report), 0 means no total column and instead a total field will be added

The following calculation might be used in the initiation calculation of a report which contains a dataset ttTest. The dataset ttTest can be any type of temp table generated and prepared for a report.

```
method('syuds.pivot','Test','sumtype=""','acctcode,acctname','kco','acctturn,acctpay',2,1);
```



Suppose we had the following data :

Table ttTest

Kco	Acctcode	Acctname	Acctturn	Acctpay	Sumtype
1	A	Supplier A	100	50	
2	A	Supplier A	150	0	
3	A	Supplier A	200	100	
	A	Supplier A	450	150	TOT
1	B	Supplier B	300	300	
	B	Supplier B	300	300	TOT

Running the pivot method above would result in three temp tables being created.

Table	Description
ttTestPage	containing page and column information
ttTestRow	containing row data
ttTestCol	containing column data



Table ttTestPage

iPageSequence	iColumn	cColumnLabel_1	cColumnLabel_2	bUsed_1	bUsed_2	bFirst	bLast	Kco_1	Kco2
1	1	1	2	Yes	Yes	Yes	No	1	2
2	3	3	Total	Yes	Yes	No	Yes	3	?

Table ttTextCol

iColumn	iPageSequence	iPageColumn	cColumnLabel	bTotalColumn	Kco
1	1	1	1	No	1
2	1	2	2	No	2
3	2	1	3	No	3
4	2	2	4	Yes	?

Table: ttTestRow

Acctnum	Acctname	iCount_1	iCount_2	iCount_3	iCount_4	Acctturn_1	Acctturn_2	Acctturn_3	Acctturn_4	Acctpay_1	Acctpay_2	Acctpay_3	Acctpay_4
A	Supplier A	1	1	1	3	100	150	200	450	50	0	100	150
B	Supplier B	1	0	0	1	300	0	0	300	300	0	0	300

If the total options is set to zero (no) then the final array entry is suppressed and separate total fields are created iCountTotal, AcctTurnTotal and AcctPayTotal with the values as you would expect.

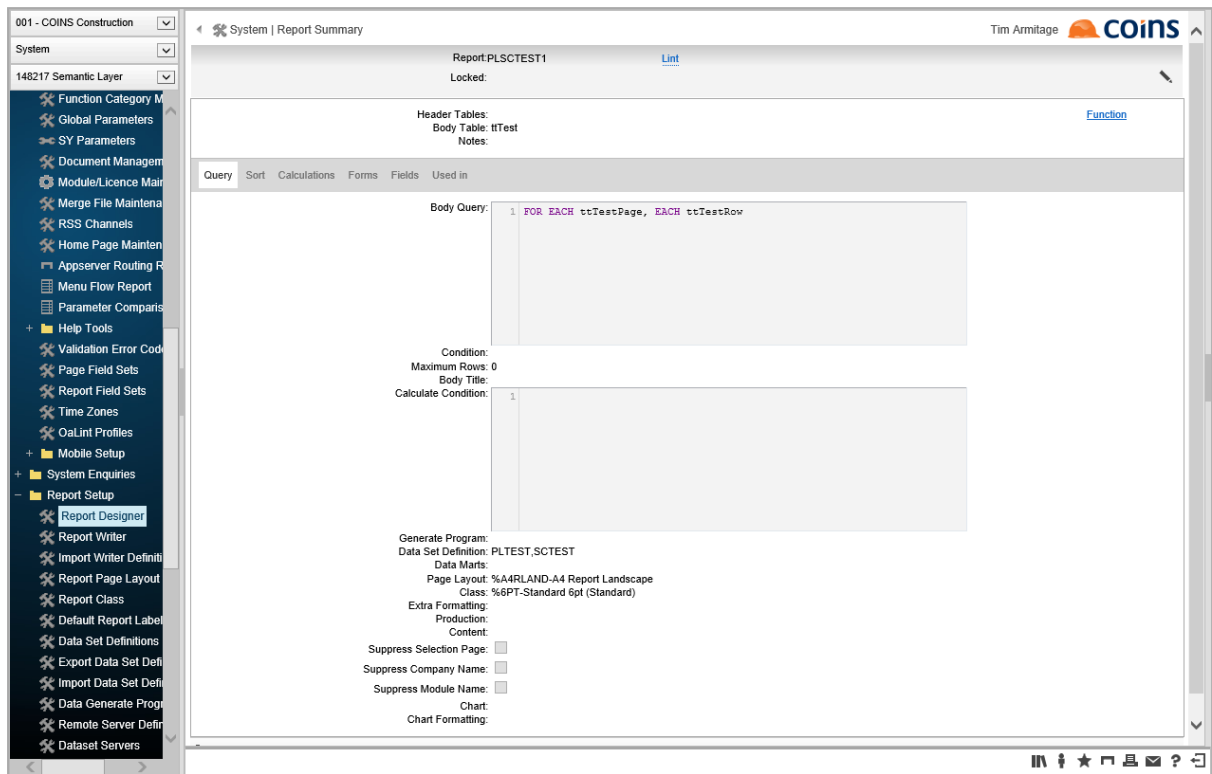
The records with SumType=TOT would not be processed because they were excluded with the query condition.

If a page size of 0 is specified then no paging will take place and a single ttPage record (iPageSequence=1) will be produced with the extent of the value fields being the same as the number of columns.

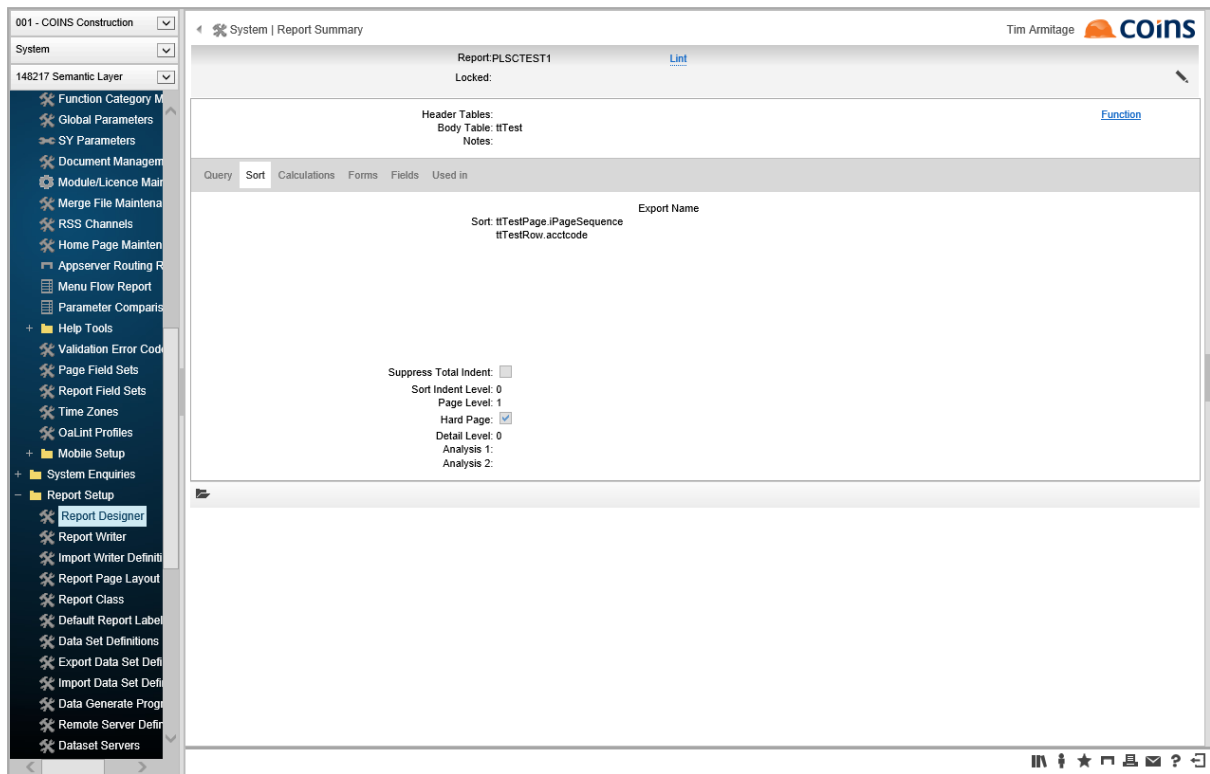
**Report Design**

The temp tables (ttTestRow and ttTestPage) produced are designed to be used in OA designer and there are supporting methods and techniques to allow a matrix report to be built.

The query used on the report should be FOR EACH ttTestPage, EACH ttTestRow. This will repeat all the rows on each page sequence (assuming there are multiple sequences of pages for all the columns to be fitted on).



The sort order should include the page sequence at the top to allow the pages to come out in the correct sequence.



You should apply a hard page to the iPageSequence level.

The fields of the report should be set out as you require using fields from the ttTestRow temp table. The columns of the matrix can be specified using {ttTestPage.iColumn\_n} where n is the page column number. The columns can be labeled using {ttPage.cColumnLabel\_n} where n is the page column number. The total column will have a label of "TOTAL" in this field. The column fields are also replicated in ttPage.cKco\_n except that the total column will have a ? value in it.



001 - COINS Construction

System

148217 Semantic Layer

- Function Category M
- Global Parameters
- SY Parameters
- Document Managem
- Module/Licence Mail
- Merge File Maintena
- RSS Channels
- Home Page Mainten
- Appserver Routing R
- Menu Flow Report
- Parameter Comparis
- Help Tools
- Validation Error Cod
- Page Field Sets
- Report Field Sets
- Time Zones
- OaLint Profiles
- Mobile Setup
- System Enquiries
- Report Setup
  - Report Designer
  - Report Writer
  - Import Writer Definit
  - Report Page Layout
  - Report Class
  - Default Report Label
  - Data Set Definitions
  - Export Data Set Defi
  - Import Data Set Defi
  - Data Generate Prog
  - Remote Server Defir
  - Dataset Servers

Query Sort Calculations Forms Fields Used in

Field	Label	Width	Height	Function	View As	Append	Hidden
ttTestRow.acctcode	Account	12mm				<input type="checkbox"/>	<input type="checkbox"/>
ttTestRow.acctname	Name	45mm				<input type="checkbox"/>	<input type="checkbox"/>
ttTestRow.acctturn_1 {ttTestPage.cColumnLabel_1}	Turnover {ttTestPage.cColumnLabel_1}	20mm				<input type="checkbox"/>	<input type="checkbox"/>
ttTestRow.acctpay_1 {ttTestPage.cColumnLabel_1}	Payments {ttTestPage.cColumnLabel_1}	20mm				<input type="checkbox"/>	<input type="checkbox"/>
ttTestRow.acctturn_2 {ttTestPage.cColumnLabel_2}	Turnover {ttTestPage.cColumnLabel_2}	20mm				<input type="checkbox"/>	<input type="checkbox"/>
ttTestRow.acctpay_2 {ttTestPage.cColumnLabel_2}	Payments {ttTestPage.cColumnLabel_2}	20mm				<input type="checkbox"/>	<input type="checkbox"/>
ttTestRow.acctturn_3 {ttTestPage.cColumnLabel_3}	Turnover {ttTestPage.cColumnLabel_3}	20mm				<input type="checkbox"/>	<input type="checkbox"/>
ttTestRow.acctpay_3 {ttTestPage.cColumnLabel_3}	Payments {ttTestPage.cColumnLabel_3}	20mm				<input type="checkbox"/>	<input type="checkbox"/>

Layout

Column Spans: 0  
Label Width:   
Alignment: Default  
Format:   
Class:   
Style:   
Extra Formatting:   
Build: syuds.buildColumnUsed\*TESTJ2  
Generate:

Data

Populate:   
Total:   
Total Types:

Calculation

Calculation: 1

Other

New Page:  Condition:   
Hide Repeat:  Repeat Field:   
Export ID:   
Chart X:   
Chart Y:   
Calculation Total: Total Displayed Values

A build condition should be used on the column fields so that they are not built when not used. The syuds.buildColumnUsed condition takes two parts to the parameter. The table name TEST (without the tt) and the page column number.

### Sample Output

With a total field

**Purchase Ledger - PLSC Combined1**  
**COINS Construction**


Account	Name	Turnover 1	Payments 1	Turnover 2	Payments 2	Total Turnover	Total Payments
AI	Abba Sealants Ltd	0.00	0.00	0.00	0.00	0.00	0.00
AI	Ashcroft Contract Servicesaa	0.00	0.00	0.00	0.00	0.00	0.00
AI0	A S E Plaster Ltd	1,988.00	57.75	0.00	0.00	1,988.00	57.75
AI000	Alsport Builders MerchantsX	14,000.00	10,829.10	0.00	0.00	14,000.00	10,829.10
AI000	Anderson & Sub	1,050.00	0.00	0.00	0.00	1,050.00	0.00
AI001	Atchil Contractors & Plant Hire	15,650.00	16,000.00	0.00	0.00	15,650.00	16,000.00
AI001	Abba Tilng Ltd*So	9,762.50	0.00	0.00	0.00	9,762.50	0.00
AI002	ASD Andrews Brown Cast Steel	112,040.00	0.00	0.00	0.00	112,040.00	0.00
AI002	Invoicing AI002	0.00	0.00	0.00	0.00	0.00	0.00
AI003	AI7.16	4,011.00	0.00	0.00	0.00	4,011.00	0.00
AI003	Authenticated receipt AI003	0.00	0.00	0.00	0.00	0.00	0.00
AI004	Alan Roofing Ltd**	0.00	2,176.65	0.00	0.00	0.00	2,176.65
AI004	ARC Quarry Products Ltd	280.00	0.00	0.00	0.00	280.00	0.00
AI005	Pro East Scotland Oil Supply	565.00	196.00	0.00	0.00	565.00	196.00
AI007	Carter Inc	0.00	0.00	0.00	0.00	0.00	0.00
AI008	.Carter Inc	1,219.00	1,780.00	0.00	0.00	1,219.00	1,780.00
AI008	.Carter Inc 2	0.00	680.00	0.00	0.00	0.00	680.00
AI01	A S E Controls (s)	0.00	0.00	0.00	0.00	0.00	0.00
AI1	A S J Beveridge Ltd	-100.00	0.00	0.00	0.00	-100.00	0.00
AI1	Andrews Ltd	0.00	0.00	0.00	0.00	0.00	0.00
AI100	Aliscott Concrs & Eng Serv Ltd	0.00	0.00	0.00	0.00	0.00	0.00
AI100	AT S Scotland Tyre Service	0.00	0.00	0.00	0.00	0.00	0.00
AI101	Aliscott Concrs & Eng Serv Ltd	255.00	100.00	0.00	0.00	255.00	100.00
AI11	A Proctor (Insulation) Ltd	0.00	0.00	0.00	0.00	0.00	0.00
AI178	Apex Industrial Ltd	0.00	0.00	0.00	0.00	0.00	0.00
AI181	John Consulting	0.00	0.00	0.00	0.00	0.00	0.00
AI2	A S S Scotland	0.00	0.00	0.00	0.00	0.00	0.00
AI234	Abba Fencing Ltd.	0.00	0.00	0.00	0.00	0.00	0.00
AI22*06	ASD Andrews Brown Cast Steel	0.00	0.00	0.00	0.00	0.00	0.00






With a total column

**Purchase Ledger - PLSC Combined1**  
**COINS Construction**



Account	Name	Turnover 1	Payment 1	Turnover 2	Payment 2	Total turnover	Total Payment
A1	Acob Seabra & Ltd	0.00	0.00	0.00	0.00	0.00	0.00
A1	Arcor Contract Services	0.00	0.00	0.00	0.00	0.00	0.00
A10	A & E Pierce Ltd	1,298.00	57.75	0.00	0.00	1,298.00	57.75
A1000	Algorit Builders Merchants	14,000.00	10,822.10	0.00	0.00	14,000.00	10,822.10
A1000	Alford & Sid	1,050.00	0.00	0.00	0.00	1,050.00	0.00
A1001	ADH Contract & Paint Hire	15,850.00	16,000.00	0.00	0.00	15,850.00	16,000.00
A1001	Alba Tiling Ltd's Co	9,762.50	0.00	0.00	0.00	9,762.50	0.00
A1002	ASD Andrew Brown CartSteel	112,040.00	0.00	0.00	0.00	112,040.00	0.00
A1002	Invoicing A1002	0.00	0.00	0.00	0.00	0.00	0.00
A1003	A1116	4,011.00	0.00	0.00	0.00	4,011.00	0.00
A1003	At the table receipt A1003	0.00	0.00	0.00	0.00	0.00	0.00
A1004	Alan Roofing Ltd 'Z'	0.00	2,176.88	0.00	0.00	0.00	2,176.88
A1004	ARC Quarry Products Ltd	260.00	0.00	0.00	0.00	260.00	0.00
A1005	Arco East Scotland Oil Supply	965.00	196.00	0.00	0.00	965.00	196.00
A1007	Cartel Inc	0.00	0.00	0.00	0.00	0.00	0.00
A1008	Cartel Inc	1,219.00	1,700.00	0.00	0.00	1,219.00	1,700.00
A1008	Cartel Inc 2	0.00	680.00	0.00	0.00	0.00	680.00
A101	A & E Contract (S)	0.00	0.00	0.00	0.00	0.00	0.00
A11	A & J Besseidge Ltd	-100.00	0.00	0.00	0.00	-100.00	0.00
A11	Andrew Ltd	0.00	0.00	0.00	0.00	0.00	0.00
A1100	Alcott Contract & Eng Serv Ltd	0.00	0.00	0.00	0.00	0.00	0.00
A1100	ATS Scotland Tyre Service	0.00	0.00	0.00	0.00	0.00	0.00
A1101	Alcott Contract & Eng Serv Ltd	255.00	100.00	0.00	0.00	255.00	100.00
A111	A P Robb (H & B) Ltd	0.00	0.00	0.00	0.00	0.00	0.00
A1175	Aper Industrial Ltd	0.00	0.00	0.00	0.00	0.00	0.00
A1181	John Courtney	0.00	0.00	0.00	0.00	0.00	0.00

**Purchase Ledger - PLSC Combined1**  
**COINS Construction**



Account	Name	Turnover Total	Payment Total	Total turnover	Total Payment
A1	Acob Seabra & Ltd	0.00	0.00	0.00	0.00
A1	Arcor Contract Services	0.00	0.00	0.00	0.00
A10	A & E Pierce Ltd	1,298.00	57.75	1,298.00	57.75
A1000	Algorit Builders Merchants	14,000.00	10,822.10	14,000.00	10,822.10
A1000	Alford & Sid	1,050.00	0.00	1,050.00	0.00
A1001	ADH Contract & Paint Hire	15,850.00	16,000.00	15,850.00	16,000.00
A1001	Alba Tiling Ltd's Co	9,762.50	0.00	9,762.50	0.00
A1002	ASD Andrew Brown CartSteel	112,040.00	0.00	112,040.00	0.00
A1002	Invoicing A1002	0.00	0.00	0.00	0.00
A1003	A1116	4,011.00	0.00	4,011.00	0.00
A1003	At the table receipt A1003	0.00	0.00	0.00	0.00
A1004	Alan Roofing Ltd 'Z'	0.00	2,176.88	0.00	2,176.88
A1004	ARC Quarry Products Ltd	260.00	0.00	260.00	0.00
A1005	Arco East Scotland Oil Supply	965.00	196.00	965.00	196.00
A1007	Cartel Inc	0.00	0.00	0.00	0.00
A1008	Cartel Inc	1,219.00	1,700.00	1,219.00	1,700.00
A1008	Cartel Inc 2	0.00	680.00	0.00	680.00
A101	A & E Contract (S)	0.00	0.00	0.00	0.00
A11	A & J Besseidge Ltd	-100.00	0.00	-100.00	0.00
A11	Andrew Ltd	0.00	0.00	0.00	0.00
A1100	Alcott Contract & Eng Serv Ltd	0.00	0.00	0.00	0.00
A1100	ATS Scotland Tyre Service	0.00	0.00	0.00	0.00
A1101	Alcott Contract & Eng Serv Ltd	255.00	100.00	255.00	100.00
A111	A P Robb (H & B) Ltd	0.00	0.00	0.00	0.00
A1175	Aper Industrial Ltd	0.00	0.00	0.00	0.00
A1181	John Courtney	0.00	0.00	0.00	0.00
A12	A G S Scotland	0.00	0.00	0.00	0.00
A1234	Alba Feeding Ltd	0.00	0.00	0.00	0.00
A123456	ASD Andrew Brown CartSteel	0.00	0.00	0.00	0.00
A1235	Alex Lawrie Fabric Ltd	0.00	0.00	0.00	0.00
A1325	Adrian Rennie & Co. Ltd	0.00	0.00	0.00	0.00
A1347	Abacus Plaster Ltd	0.00	0.00	0.00	0.00
A1368	F L Walker & Co G A	0.00	0.00	0.00	0.00
A1419	Andrew Walker Ltd	0.00	0.00	0.00	0.00
A1476	Alight Decorating Co Ltd	0.00	0.00	0.00	0.00
A1500	ACS Environmental Ltd	0.00	0.00	0.00	0.00

#### 7.13.8. Repeat

This method removes repeated fields from a temp table. Typical usage is if the data set contains header and detail data and the repeating header details only want to be shown (and aggregated) once. This method will allow you to manipulate the dataset so that repeated values only appear once.

```
Method('syuds.repeatSource','Source','kco,job_num');  
Method('syuds.repeatFields','1','kco,coc_name');  
Method('syuds.repeatFields','2','job_num,job_name,RO_job_costs^TD');  
Method('syuds.repeatExec');
```

The ttSource table will be replaced with a copy where the repeating fields kco, coc\_name and job\_num,job\_name are shown once per kco and job\_num sort sequence.

repeatSource specifies the source table and the sort fields to be used.

repeatFields specifies for each of the sort sequence the fields that should be shown just once

In this example RO\_job\_costs^TD (and job\_num and job\_name) are shown only once for each job\_num within kco. This would allow this column to be aggregated on the report.

repeatExec actually performs the repeat field blanking and saves the temp table overwriting the original source data.

#### 7.13.8. \$yuds.Store

This option is identical to the data mart writing on the tail end of a report except that you control the date/time of the extract.

The datamart must be defined in the usual way and the mapping of the fields in the datamart to the fields in the dataset must be configured.

For example

```
Method('syuds.store',100,'job,cost','JOBDM,COSTDM',TODAY,0);
```

Would store the dataset ttJob in datamart JOBDM and similarly dataset ttCost in datamart COSTDM with an extract date of midnight on the day of running in company 100. The extract date would be expected to be a report input/selection value.

#### 7.13.8. ~~Sum~~ ~~Sum~~

This method will combine and sum rows from two or more dataset in to one or more new datasets.

Suppose we have a dataset ttCostRev

Kco	Job_num	dCosts	dRev
100	1000	100	150
100	1001	200	250
100	1002	0	100
100	1003	100	0

Then the resulting dataset ttKcoSum

Kco	dCosts	dRev
100	400	500

would be produced with the following commands

```
Method('syuds.sumSource','CostRev','');  
Method('syuds.sumTable','KcoSum','kco','','dCosts,dRev');  
Method('syuds.sumExec');
```

SumSource specifies the input dataset and a condition to apply to that set of records.

sumTable specifies the output summary required. The first parameter is the output dataset name (without the tt prefix), the second parameter is the key fields, the third parameter is other fields to be assigned (similar to key fields but not used to find uniqueness e.g. keys=kco, fields=coc\_name), the fourth parameter is the fields to sum.

sumExec executes the summing and creates the required output datasets. Multiple sumTable methods may be used to create multiple summaries on a single pass through the source data.

A short version with defaults exists

```
Method('syuds.sum','CostRev','','KcoSum','kco','','dCosts,dRev');
```



With the following parameters input table, condition, output table, keys, fields, sum fields.

#### 7.13.8. ~~T~~TableAlias

```
Method('syuds.tableAlias','source','output');
```

This takes a table and renames it for the purposes of using a table multiple times.

1<sup>st</sup> parameter is the source dataset table and the 2<sup>nd</sup> parameter is the new dataset table.

### 7.13.8. ~~52~~syuds.TimeSlice()

This is a new method in syuds.p to manipulate an input dataset or datasets in to a new summarized dataset which has a date/time element.

The method can be called like any other post processing method on a dataset, for example

```
method('syuds.timeslice','FOR EACH ttStats','TimeStats',{fromtime},{totime},{interval},  
{timeunit}','moe_key1,moe_key2','moc_snapshot',')
```

The parameters are as follows:

- 1 Query      The query to be run against existing dataset(s) which will return a set of record containing the data that is to be time sliced
- 2 TableName The output table name
- 3 From      A string containing a valid from date or date/time (depending on the units Date/Time being used). If omitted then the time range will not be completed and records will only exist for the data being processed
- 4 To         Same as From date/time Date/Time
- 5 Interval   An integer value to be used in conjunction with the time unit field to allow the creation of the complete set of time records
- 6 Time Unit  “S”econds (for date/time), “D”ay, “W”eek, “M”onth, “Y”ear
- 7 Keys       The key fields used to summarise the data. Can be left blank and just the time element will be used
- 8 Time Key   The field in the source data that contains the date/time field to be used to slice the data
- 9 Fields     The fields from the source data to be aggregated. If left blank then ALL decimal fields in the source tables will be aggregated.

For each field that is aggregated the total, max, min and average values for the time slice will be evaluated.

If the Time Unit field is “S” for seconds then the Interval is the number of seconds for each time slice and the start and end date/time values are date/time format strings. E.g.  
From=15/01/16 08:00&To=15/01/16 18:00&Interval=3600 would produce hourly slices from 8am until 6pm.

If the Time Unit field is “D” for days then the interval is not used and single day records are produced between the from and to dates.

If the Time Unit field is “W” for weeks then the interval indicates the day of the week for the week ending i.e. 1 for Sunday to 6 for Saturday. Week end dates for that day are then produced between the start and end dates.

If the Time Unit is “M” for months then the interval is ignored and calendar month end dates are used between the start and end dates.

If the Time Unit is “Y” for years then the interval is ignored and calendar year end dates are used between the start and end dates.

The results dataset will contain four fields for each of the aggregated fields (one of the same name for the total and then extensions min, max and avg for the other three values). A count is always added plus the unique grouping key and key date/time.



#### 7.13.8. Top

This method returns the Top n records within a dataset when a sort is defined.

```
Method('syuds.top','FullList','drev>100000','by drev descending',10,'Top10Rec');
```

The 1<sup>st</sup> parameter is the source temp-table. The 2<sup>nd</sup> parameter is the condition applied to the query, the 3<sup>rd</sup> parameter is the sort order, the 4<sup>th</sup> parameter is the number of records returned , the 5<sup>th</sup> parameter is the output table name.

In the example above it is taking all the records from the ttFullList table and for records which revenue exceeds 100000 then it sorts it by the revenue field (highest to lowest) and returns the top 10 records into a table called ttTop10Rec.

#### 7.13.8.1 Union

This method will combine rows from two or more dataset in to a new dataset.

Suppose we have a dataset ttCost

Kco	Job_num	dCosts
100	1001	100
100	1002	150

And a dataset ttRev

Kco	Job_num	dRev
100	1001	200
100	1002	250

Then the resulting dataset (ttCostRev) might be

Kco	Job_num	dCosts	dRev
100	1001	100	0
100	1002	150	0
100	1001	0	200
100	1002	0	250

This would be achieved with the following method calls.

```
Method('syuds.unionFields','kco,job_num,dCosts,dRev');  
Method('syuds.unionTable','cost','kco,job_num,dCosts,');  
Method('syuds.unionTable','rev','kco,job_num,,dRev');  
Method('syuds.unionExec','CostRev');
```

unionFields specifies the fields in the returned dataset.

unionTable specifies the source dataset to combine. The first parameter is the dataset name (without the tt prefix), the second parameter is a condition for the records to select from this source dataset, the third parameter is the fields to combine.

unionExec specifies the output dataset and executes the union.

If the fields specified are the same then the values are combined in to the same field

e.g.

```
Method('syuds.unionFields','kco,job_num,dValue');  
Method('syuds.unionTable','cost','kco,job_num,dCosts');  
Method('syuds.unionTable','rev','kco,job_num,dRev');  
Method('syuds.unionExec','CostRev');
```

Would produce the output dataset

Kco	Job_num	dValue
100	1001	100
100	1002	150
100	1001	200
100	1002	250

There is also a short version with default options.

```
Method('syuds.union','cost,rev','kco,job_num,dCosts,dRev','CostRev');
```

ttCost and ttRev would be combined with fields kco,job\_num,dCosts,dRev in to dataset CostRev. This is equivalent to the first example above.

## 8 Workflow Overview

People are most productive when they are doing the things they are good at – thinking, communicating, creating. They are least productive and most prone to error when they are involved in boring, repetitive processes. Just the opposite is true of systems.

Businesses are therefore most efficient when they have the three key resources of people, processes, and systems properly balanced, integrated, and focused on their strengths:

People developing and implementing processes and new ways of working.

Processes that optimise quality, consistency, timeliness, and governance/audit.

Systems that facilitate the way people work by automating the core processes.

COINS Workflow brings the benefits of this integration to construction businesses. By using workflows within COINS, customers can automate routine but essential processes such as invoice approval, employee expense processing, purchase orders, and site requisitions. This frees costly people resources to focus on improving and managing the business.

### 8.1 Business Benefits

Because of its flexibility, the applications and benefits of COINS Workflow are limited only by your imagination. Key business benefits include:

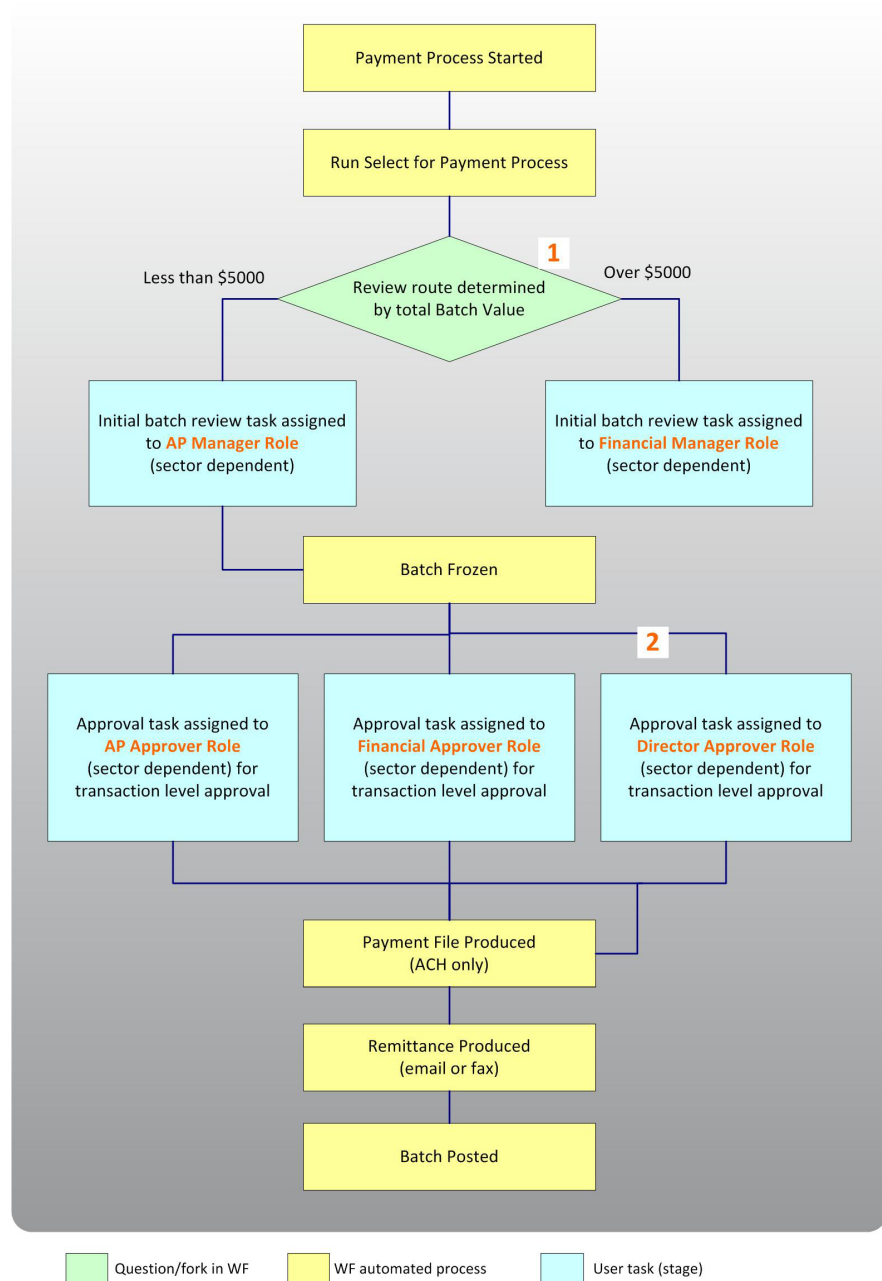
- Process cost reduction.
- Improved productivity for professional staff, who can spend less time on repetitive processes and more on managing the business.
- Improved process resilience and reduced dependence on individuals.
- Enhanced environmental responsibility through reduced reliance on paper.
- Enhanced audit trail.

## 8.2 The COINS Workflow toolkit

Whether COINS workflow is configured by consultants from the Business Intelligence Team or by customers' own technical staff, the same toolkit is available. Its key features include:

- Triggering workflow processes by incoming documents, events within the COINS database (such as a value being exceeded), or manually.
- Multiple workflow branches to allow parallel processes, optionally
- Merging back together once each branch has completed. Conditional branching enables the implementation of complex authorization levels and procedures.
- Workflow tasks and stages assigned to individuals or groups.
- User alerts communicated by email, SMS text message, or through a COINS workbench on a PC or a PDA.
- Full integration with all COINS modules, with access to every database table and function.
- User management of tasks and events through the COINS Activity Workbench, with links directly to the COINS item or transaction to be processed.
- Integration with Microsoft® Outlook®, including linking an email back to a COINS transaction.
- Full security provided through the COINS Business Logic Layer.
- Full audit trail of all workflow activities, with tracking of all current and completed workflows.
- Automatic escalations managed with defined task duration.
- Delegation of workflow actions during vacations or other absences, with expiration dates.
- Integrated Document Management with the ability to read data from scanned documents into workflow processes and COINS input screens, avoiding manual processing. Document Management items can be created automatically from COINS alerts, emails and SMS messages.

## 8.3 Example Workflow

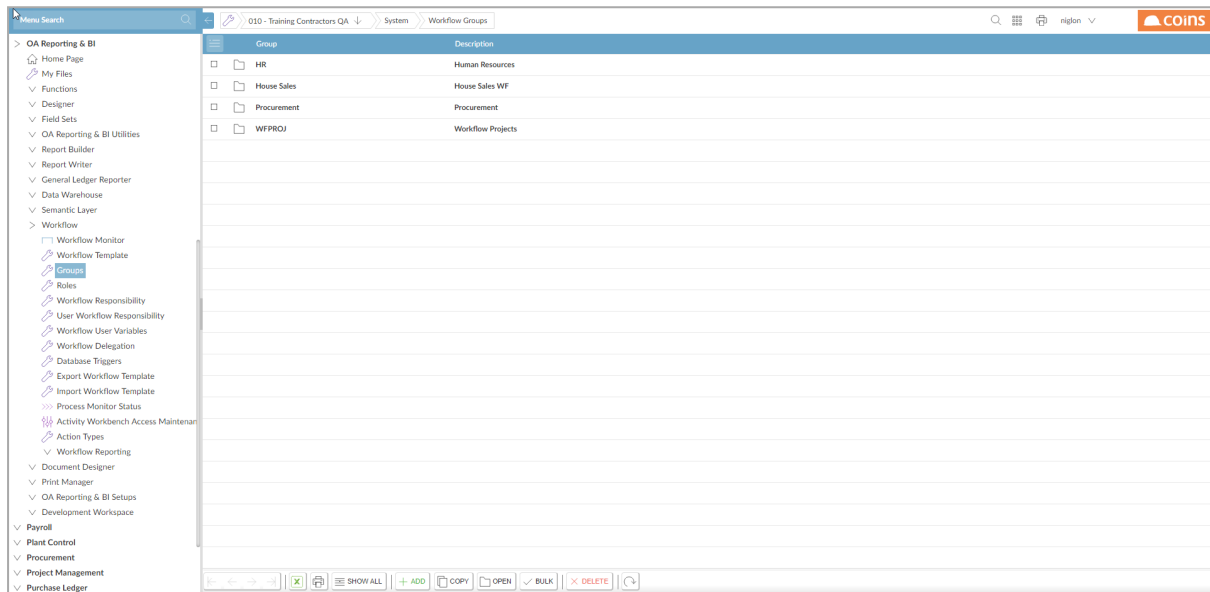


- 1** This could be a check on total cash value of the payment run, to ensure that payments are only released where there are sufficient cash reserves.
- 2** This could be a multi-layered process based on individual payment value, or a fork where approval is requested from more than one source. For example, you could require a double signature or a CEO signature if a single payment exceeds a threshold.

### 8.3.1

## 8.4 Workflow Groups

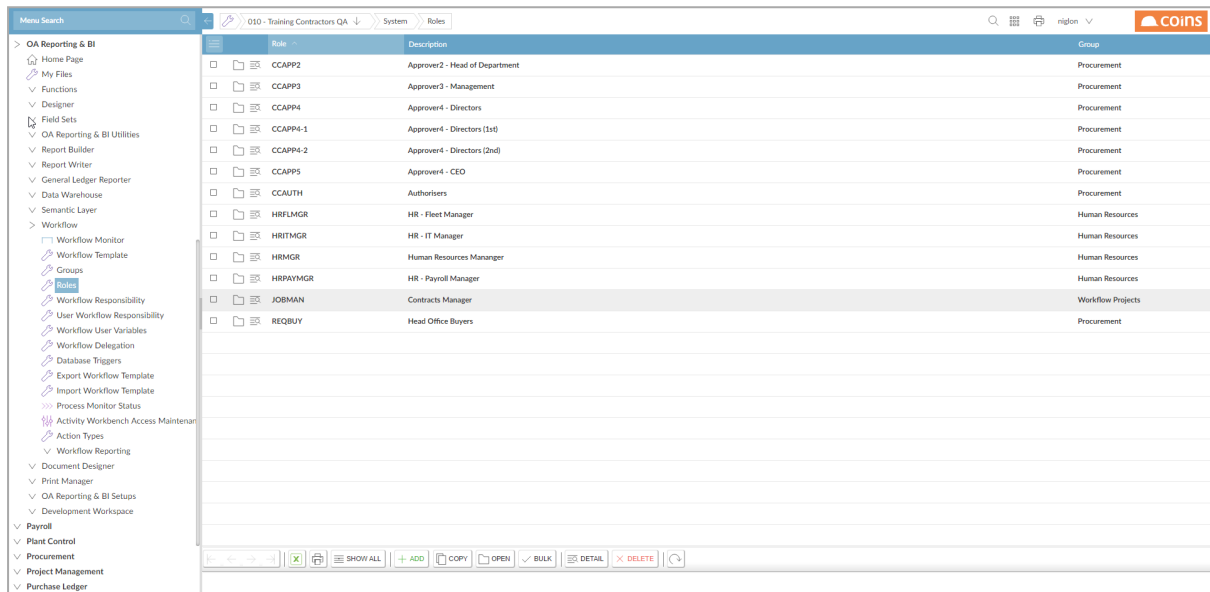
Workflow groups are simply a means to organise Workflows into business areas, for example Procurement, HR, Purchase Ledger etc. At a later stage you can specify which roles belong to a group; and if you then assign a Workflow to a group, any member of the group can action it (see Workflow Roles and Users).



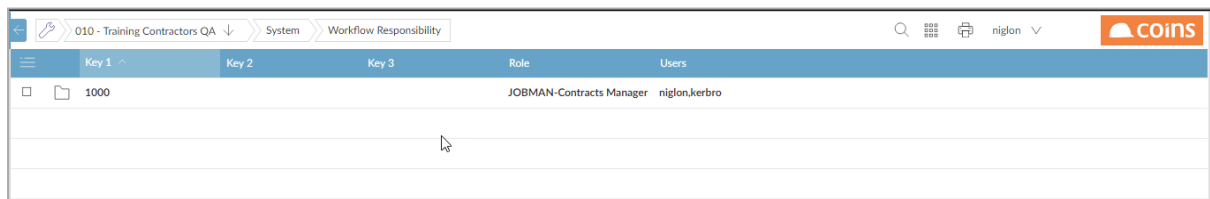
Field	Description
Group	The workflow group code. Use this code to group roles and workflow templates.
Description	The description of the workflow group.

## 8.5 Workflow Roles

As part of the design of a workflow, it is necessary to define who in an organisation is going to be responsible for all or part of the workflow processes. Typically a role will fall to a business role such as a Financial Manager or Purchase Ledger Supervisor, or indeed a team of people such as a Buying Team.



Create a Workflow Role by assigning a code and description. The workflow roles will require a responsibility type and where appropriate a key or keys.



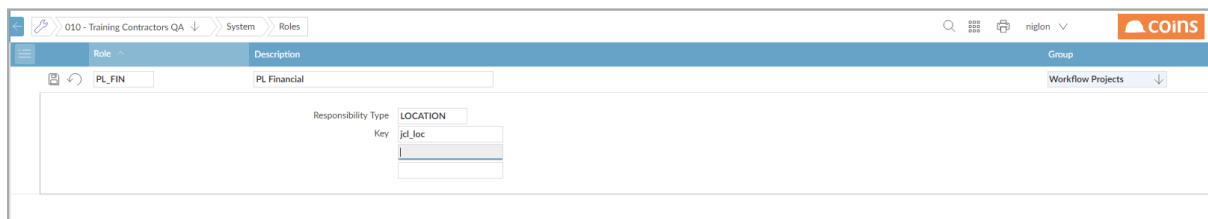
Field	Description
Role	The workflow role code.
Description	The description of the workflow role
Group	The workflow group code (see Workflow Groups). Use this code to group roles and workflow templates. May be left blank
Responsibility Type	This code allows you to specify, for this role, what is the workflow going to use for routing.  DO NOT use either CONTRACT or COMPANY as a



Field	Description
	responsibility type as these are reserved words with a specific use within Workflow setup.
Key	The key that corresponds to the responsibility table keys. The keys must be defined as variables in the workflow to allow the correct allocation of people to stages.  Where the Responsibility Type relates to a group, then the key may be left blank

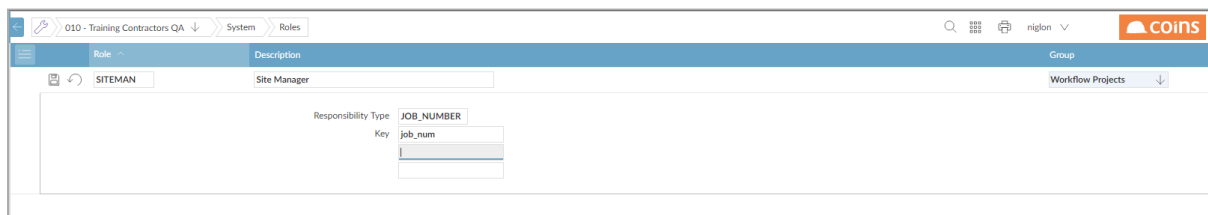
### 8.5.1 Example Roles

In this example, the Responsibility Type has been defined as LOCATION and the key has defined as jcl\_loc (Contract Location).



This will allow the workflow to determine the responsibility from the Contract Location code on a transaction.

In the second example, the role of Site Manager is defined as using Contract information to route the workflow by use of the job\_num field on a transaction.



Please remember DO NOT use either CONTRACT or COMPANY as a responsibility type as these are reserved words with a specific use within Workflow setup.

## 8.6 Workflow Responsibilities

Once Roles have been defined within the system, their responsibilities within the workflow may be configured. Workflow Responsibilities are used to convert roles set in a workflow to actual user IDs.

In Role Maintenance a role is associated with a type and up to 3 variables (These variables are defined within a workflow).

In Responsibility Maintenance a series of records are created that map possible values of these variables to one or more user IDs. When the role is used in a workflow stage and the values match then the user IDs are used.

Field	Description
Type	This code is referenced when setting up a workflow role. Use Workflow Responsibility to set up different responsibilities depending on the value of a variable defined in the workflow.
Keys 1 - 3	The appropriate values of the Role key.  The value in these key fields corresponds to the variable set in the equivalent key of the role that is linked to this responsibility (see role maintenance).  If the value of this variable set at the workflow stage where the associated role is used matches this responsibility value (along with the other two key fields) then the user ids configured on this responsibility are used in that workflow stage.
Role	The person who must action this stage
Users	The user or users who are responsible for this role in a workflow. These must be users' setup already within the COINS system.

### 8.6.1 Example Responsibilities

In this example, the role of JOBMAN has been defined in the Workflow Roles with job\_num as a key.



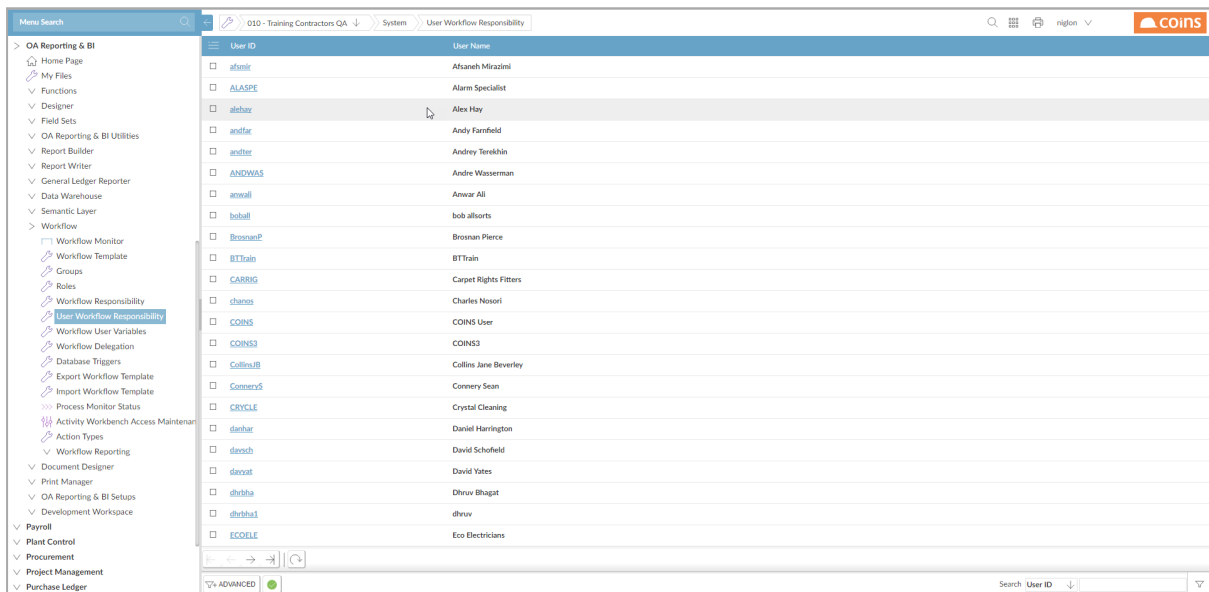
In the Responsibility setup JOBMAN is defined with the key set to 1000 and a user list entered. This indicates that a task referencing Contract 1000 will be routed to the users **niglon** and **kerbro**

The screenshot shows a web application interface for setting up workflow responsibility. The breadcrumb navigation at the top reads "010 - Training Contractors QA > System > Workflow Responsibility". The user "niglon" is logged in. The table below has columns for "Key 1", "Key 2", "Key 3", "Role", and "Users". A single row is visible with the value "1000" in the "Key 1" column, "JOBMAN-Contracts Manager" in the "Role" column, and "niglon,kerbro" in the "Users" column.

Key 1	Key 2	Key 3	Role	Users
1000			JOBMAN-Contracts Manager	niglon,kerbro

## 8.7 User Workflow Responsibilities

On occasions it is desirable to be able to find and amend all the role assignments for a user. This can be done by searching through the responsibilities maintenance screen but it is possible for COINS to do this for you. This might be useful if a user leaves the business and a new user is to be assigned to each of their role responsibilities.



The User Workflow Responsibility maintenance screen shows a list of users. Select the link on the user for which you want to maintain responsibilities. All responsibility records which contain the selected user are shown for update.



The screenshot shows a web application interface for 'User Workflow Responsibility'. The breadcrumb navigation is '010 - Training Contractors QA > System > User Workflow Responsibility'. The user is identified as 'User: afsmir' and 'Afsaneh Mirazimi'. The table has columns for 'Type', 'Key 1', 'Key 2', 'Key 3', 'Role', 'Company', and 'Users'. There are six rows of data, each with a checkbox and a folder icon in the 'Type' column. The 'Role' column contains codes like SWHACCGRP, SWHBUYAPR, SWHCONAPR, SWHDIVADM, SWHFINAPR, and SWHOVRAPR. The 'Company' column contains '10' and the 'Users' column contains various email addresses like 'niglon.suejon.afsmir'. At the bottom, there is a toolbar with buttons for 'SHOW ALL', 'OPEN', 'CONCURRENT', 'BULK', 'DELETE', and 'REGENERATE'. A search bar is also present at the bottom right.

Type	Key 1	Key 2	Key 3	Role	Company	Users
<input type="checkbox"/>				SWHACCGRP	10	niglon.suejon.afsmir
<input type="checkbox"/>				SWHBUYAPR	10	train5,niglon.suejon.afsmir
<input type="checkbox"/>				SWHCONAPR	10	train2,niglon.suejon.afsmir
<input type="checkbox"/>				SWHDIVADM	10	train1,niglon.suejon.afsmir
<input type="checkbox"/>				SWHFINAPR	10	train4,niglon.suejon.afsmir
<input type="checkbox"/>				SWHOVRAPR	10	train3,niglon.suejon.afsmir

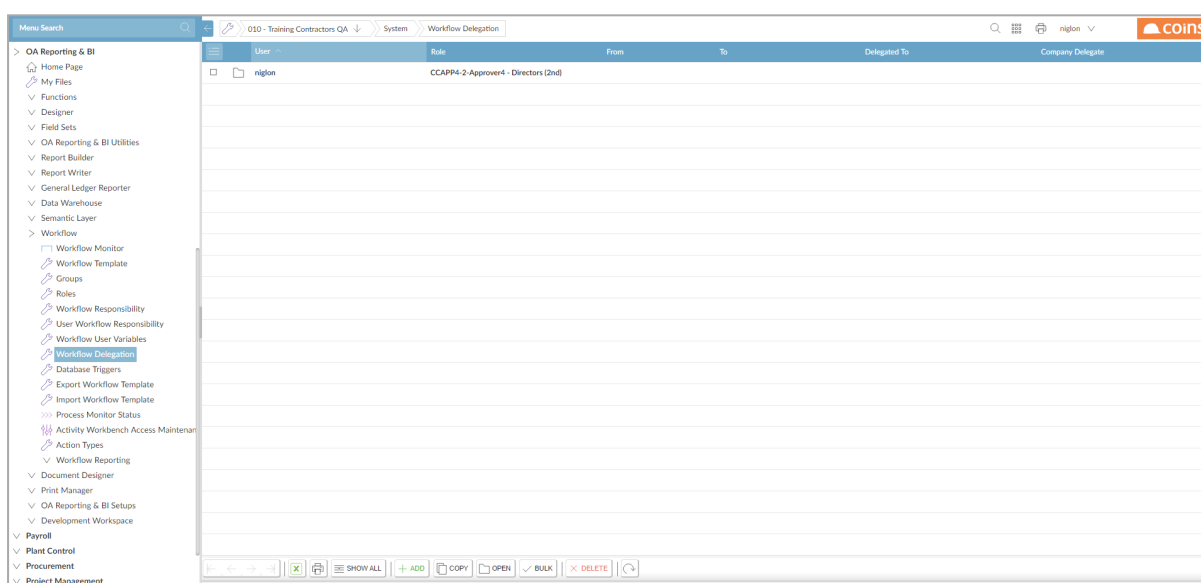
These records can now be amended. When amending the user name, please note that the new user must be licenced for workflow.

The regenerate button can be pressed to regenerate the list. The button can be used to go back to the user selection screen.

## 8.8 Workflow Delegation

Once all roles and responsibilities have been configured it is possible to delegate actions and tasks for users temporarily, for example during periods of annual leave.

When a workflow task is assigned to a user this table of delegations is consulted. If a record for the specific role exists for the user then that is used. If not then the record for all roles is used if it exists. If neither record is found then no delegation takes place and the original user is assigned the task. The date that the task is assigned is used to find the delegation record with the correct date range.



Simply define the user and role to be delegated. If the user belongs to several roles, you can enter more than one delegation record and assign each role to another user individually, or you may specify ALL and simply allocate all roles to another user.

Each delegation record may have each a unique start date. A blank start date is allowed indicating the delegation has no start date. A blank end date is also allowed indicating no end date.

Workflow Delegation is intended to stop new actions and tasks being assigned to a user who is unavailable. Any current actions and tasks will need to be reassigned manually.

A company specific delegation is also now available. If the company specific delegation is specified (in each company) then it is used. If it is blank then the global delegation is used.

The following business rules are applied:

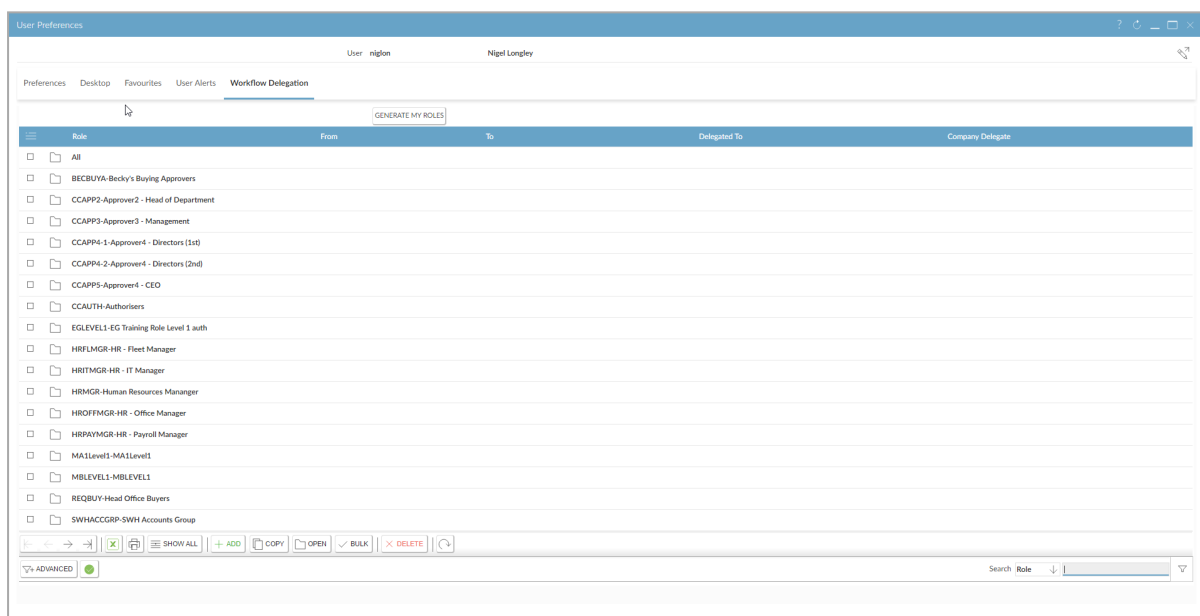
- You are not allowed to overlap from/to dates for a single combination of user and role. i.e. only one delegation can apply on a single date.
- The user must be valid and licenced for workflow
- The delegate must be valid and licenced for workflow
- The user and the delegate cannot be the same user
- The company delegate can be blank indicating the global delegate should be used.
- The global delegate can be left blank indicating that no delegation is to be done. This can be useful to leave the role records in place and just assign and remove the delegation as required.

### 8.8.1 Self-Service Workflow Delegation


When a user who takes part in a workflow is going to be absent from work they will wish to delegate the roles they fulfil to one or more other users.

This can be done in the delegation maintenance procedure described above however the delegation maintenance routine gives access to all users' delegations so is not appropriate for many users.

A function is provided that allows the user to set up and maintain delegations for themselves only. A new tab is shown on the user preferences for the user when they are logged in; if they are licenced for workflow.



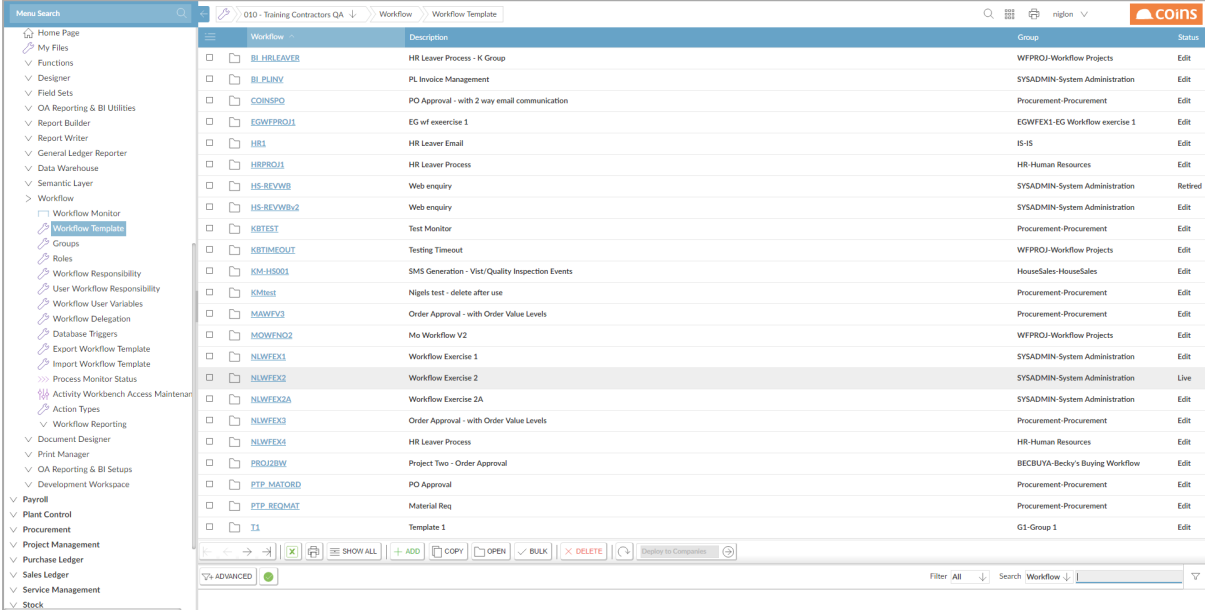
This screen shows the same information as the full delegation maintenance screen except that the user is fixed and shown in the header. The same rules apply for setting of the delegations as in the full version.

There is a button provided in the footer  that will scan all workflow assignments in all companies and build one delegation record per role. This is an aid to the user to ensure they do not miss a role. The records are created with a blank delegation (i.e. no delegation) but the record being present is a reminder to set it if required. They can be deleted or left blank if no delegation is required.



## 8.9 Workflow Templates

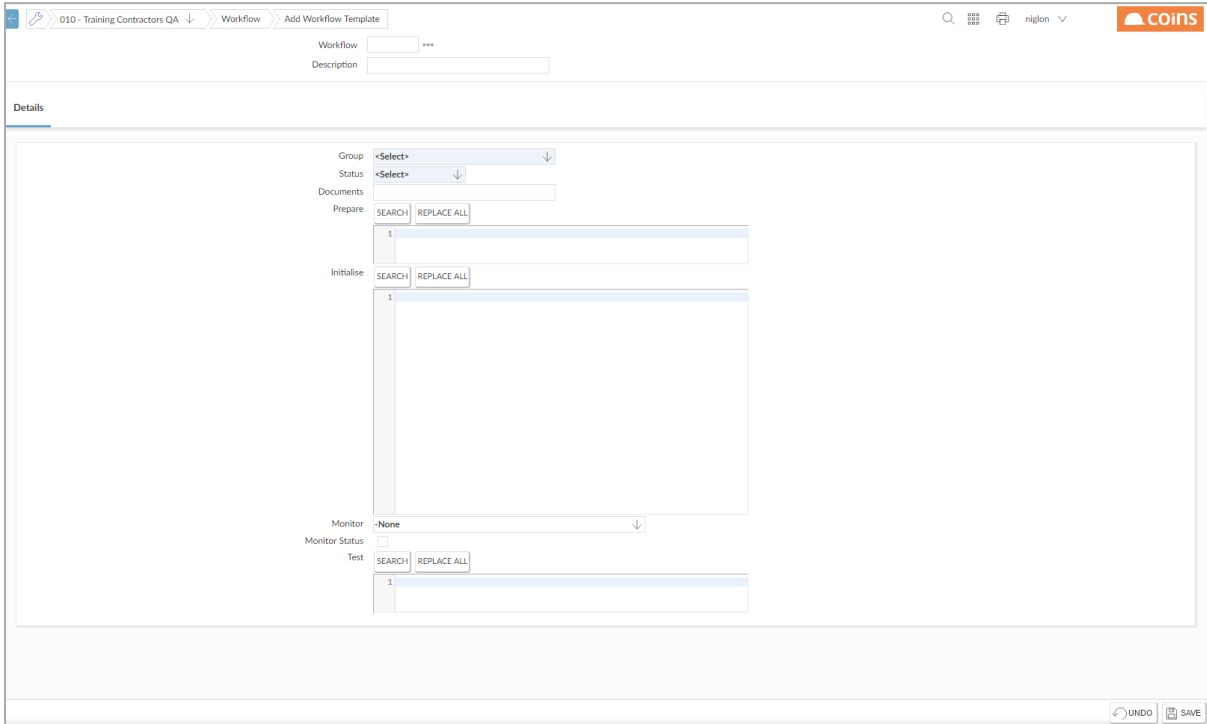
This option allows you to add and maintain the workflow definitions.



Workflow	Description	Group	Status
BI_HBLEAVER	HR Leaver Process - K Group	WFPROJ-Workflow Projects	Edit
BI_PUNV	PL Invoice Management	SYSADMIN-System Administration	Edit
COINSP0	PO Approval - with 2 way email communication	Procurement-Procurement	Edit
EGWFPROJ1	EG wf exercise 1	EGWFEX1-EG Workflow exercise 1	Edit
HRI	HR Leaver Email	IS-IS	Edit
HRPROJ1	HR Leaver Process	HR-Human Resources	Edit
HS-REVWB	Web enquiry	SYSADMIN-System Administration	Retired
HS-REVWBv2	Web enquiry	SYSADMIN-System Administration	Edit
KBTST	Test Monitor	Procurement-Procurement	Edit
KBTMOUT	Testing Timeout	WFPROJ-Workflow Projects	Edit
KM-HSQ01	SMS Generation - Visit/Quality Inspection Events	HouseSales-HouseSales	Edit
KMtest	Nights test - delete after use	Procurement-Procurement	Edit
MAWFX3	Order Approval - with Order Value Levels	Procurement-Procurement	Edit
MOWFNO2	Mo Workflow V2	WFPROJ-Workflow Projects	Edit
NLWFX1	Workflow Exercise 1	SYSADMIN-System Administration	Edit
NLWFX2	Workflow Exercise 2	SYSADMIN-System Administration	Live
NLWFX2A	Workflow Exercise 2A	SYSADMIN-System Administration	Edit
NLWFX3	Order Approval - with Order Value Levels	Procurement-Procurement	Edit
NLWFX4	HR Leaver Process	HR-Human Resources	Edit
PROJRW	Project Two - Order Approval	BECBUYA-Becky's Buying Workflow	Edit
PTP-MATORD	PO Approval	Procurement-Procurement	Edit
PTP-REQMAT	Material Req	Procurement-Procurement	Edit
T1	Template 1	G1-Group 1	Edit

Each workflow template is defined across a number of tabs

### 8.9.1 WF Template - Details Tab

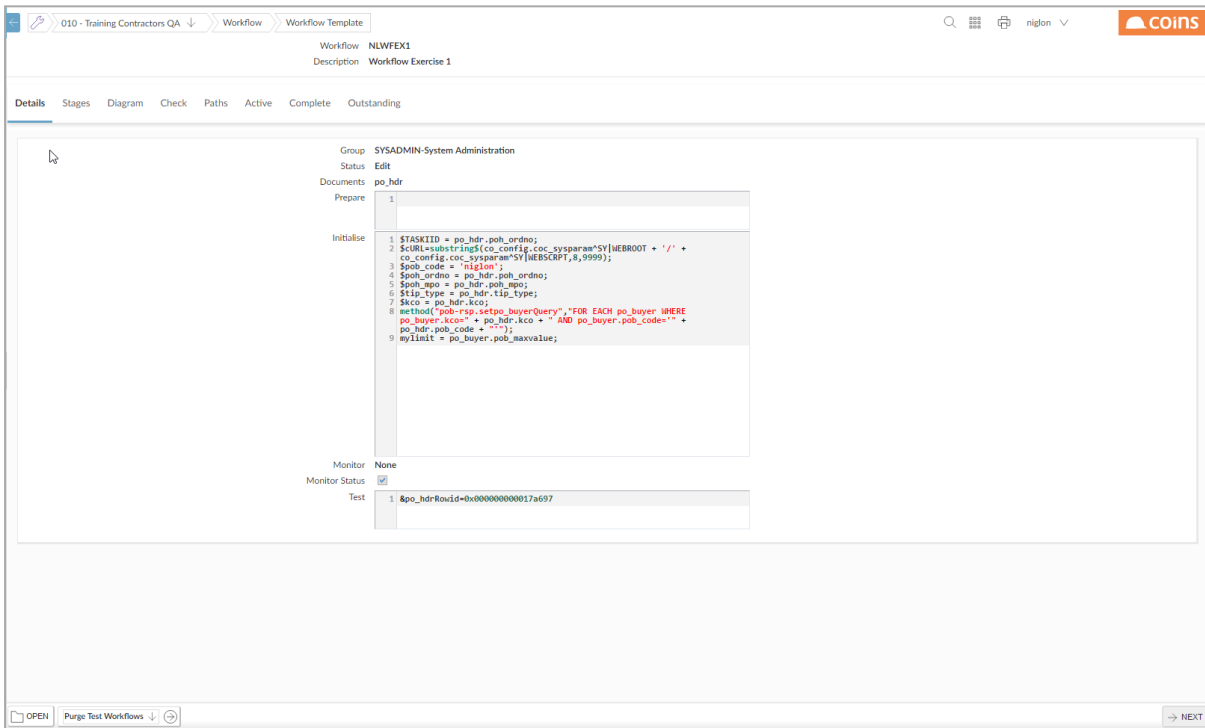


Field	Description
Workflow	Workflow template code.
Description	The workflow template description.
Group	The workflow group code. Use this code to group roles and workflow templates.

Field	Description
Status	<p>The status of the template.</p> <p>Edit - The workflow is not live; you can still edit and test it. NB. Edit has a limit of 25 active records before requiring a purge of transactions or changing the status of the workflow to Live.</p> <p>Live - The workflow is live and you can no longer edit the logical fields involved in the workflow. There is a limited number of fields which are editable.</p> <p>Retired - The workflow is no longer live leaving the active stages to continue but not allowing new workflows to be launched. Your licence limits the number of live workflows you can have. If you want to change an existing workflow, you must retire the original, copy it in edit mode and edit the copied workflow. (See Appendix A for details on Workflow Licensing).</p>
Documents	<p>The COINS table(s) that are to be associated with the workflow.</p> <p>If this is filled in, the user must select a record from this table in order to run this workflow. The workflow can be independent of any particular table, in which case this can be blank.</p>
Prepare	<p>This is a calculation that takes place prior to the document load and can be used to assign specific RSP if not that of the standard table. This would be on recommendation and advice from COINS BI Team. This is not a field that would normally be used in workflow.</p>
Initialise	<p>The initial calculations that are performed at the initiation of the workflow.</p> <p>A variable of \$TaskID must be initialised on each workflow. This will be the identifier for each instance of the workflow and is shown in the description of stage tasks, for example the internal reference of a transaction.</p> <p>The initialisation uses standard COINS calculation syntax (see associated documentation) and can also use standard COINS Methods.</p>
Monitor	<p>The user who COINS should alert with a duplicate task if a stage extends beyond the expected duration. The monitor specified on the template will be used unless there is a specific monitor specified on the stage itself. – see Workflow Monitors below for more detail.</p>
Monitor Status	<p>A check box which will indicate if the workflow is to appear on the Monitor Tab on the Workflow Monitor.</p>

Field	Description
Test	The URL/ROWID to be processed if the workflow is started for test purposes when in edit mode.

Below is an example of a workflow template; including examples of initialisation calculations which may be required and a Test Invoice which will enable to launch a test workflow.



Workflow: NLWFEX1  
Description: Workflow Exercise 1

Details | Stages | Diagram | Check | Paths | Active | Complete | Outstanding

Group: SYSADMIN-System Administration  
Status: Edit  
Documents: pob\_hdr

Prepare: 1

Initialise:

```

1 $TASKIID = pob_hdr.pob_ordno;
2 $URL=substring((co_config.coc_sysparam$V|WEBROOT + '/' +
co_config.coc_sysparam$V|WEBSRPT,8,9999));
3 pob_code = '112101';
4 $pob_ordno = pob_hdr.pob_ordno;
5 $pob_mpo = pob_hdr.pob_mpo;
6 $sta_type = pob_hdr.fip_type;
7 $kco = pob_hdr.kco;
8 method('pob-rsp.setpo_buyerQuery','FOR EACH po_buyer WHERE
po_buyer.kcc=' + pob_hdr.kcc + " AND po_buyer.pob_code=" +
po_hdr.pob_code + "'");
9 $y1limit = pob_buyer.pob_maxv1ue;

```

Monitor: None  
Monitor Status:

Test:

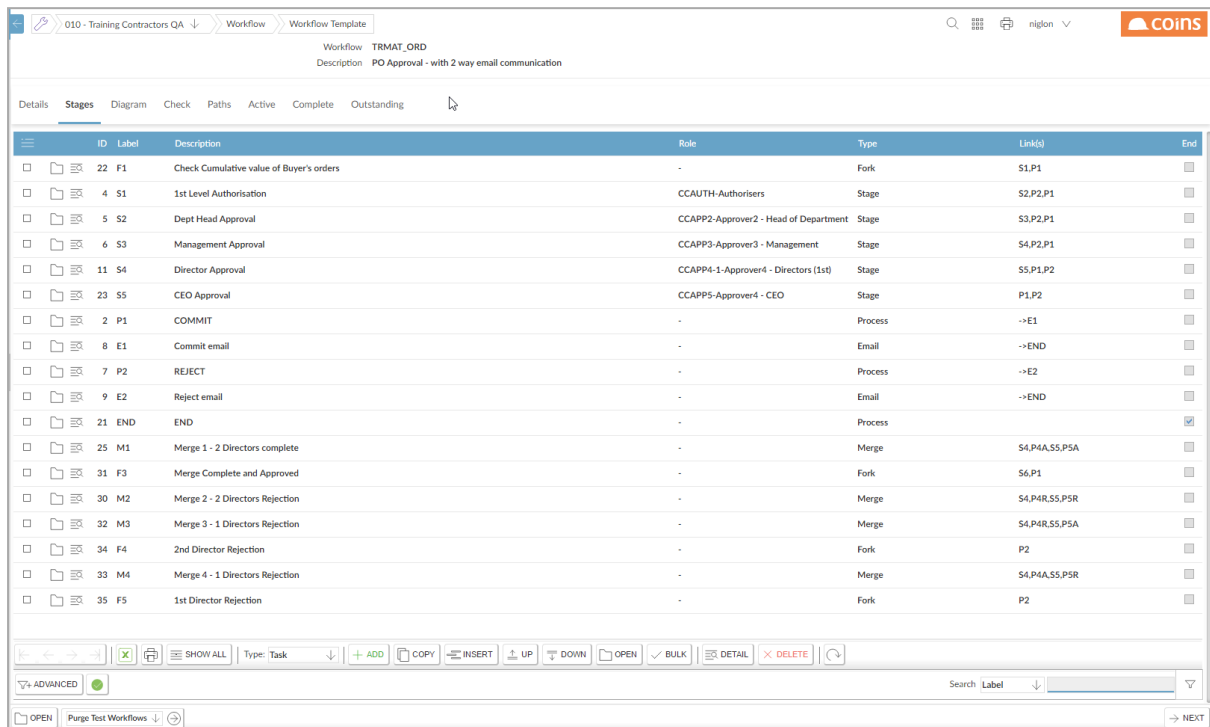
```

1 &pob_hdrRowid=0x00000000000017a697

```

OPEN | Purge Test Workflows ↓ | → NEXT

## 8.9.2 WF Template - Stages Tab



Workflow Template Stages allows you to define the stages that make up the workflow.

Field	Description
Label	Label of the stage. Used to link other stages to this one.
Description	Description of the stage. This is the text that will appear in the user's Activity Workbench, along with the general description of the workflow and the task ID.  If another stage links to this stage, this description will appear on the Next Stage list for the stage that links to it.
Role	The role that must action this stage. (You can use a role calculation to manipulate this further.)
Type	The type of stage process that should be performed (see Workflow Stage Types).
Link(s)	A list of the labels of other stages that are linked to this stage.
End	Whether this stage is an end stage. The workflow stops if it reaches an end stage.

There are currently 10 types of workflow stage available, each will require a Label and a Description. The description will appear as the description in the Activity Workbench for the stage (with the Workflow TaskID reference). The label is used to determine the links between stages. The Id which is shown on screen will be automatically assigned and is useful when debugging the workflow with the tools on the Check tab (see documentation further on in this document).

The workflow role will determine who will be assigned a stage or task in the workflow. A role can be one of three options:

Role	When a stage is assigned to a role COINS will cross reference the Role and the associated Responsibilities and assign the stage to the appropriate User(s). It will then perform the Role Calculation.
Group	The person who is handing on the task may select an individual, or assign it to everyone in the list.  When a workflow stage is assigned to a Group, the appropriate activity will be assigned to every member of the group and will appear on each member's Activity Workbench.
Individual	Any stage can simply be assigned to an individual.

If the role calculation result is a list, the user who is handing on the task must first select one individual from the list to be the person to action the task.

An individual can also be calculated from the Workflow document or 'hard-coded' into the role calculation.

If any stage is assigned to more than one user, the task will appear on each user's workbench. As soon as one user assigns their task to a status of anything other than 'Not Started' the corresponding tasks on all other users Workbenches will be completed. It will remain available for view and will be automatically updated with the progress and user of who has started the task. If they put the status back to "Not Started" then the task will be replaced on all the users in the group again.

Once any of the tasks are completed all tasks on other user's workbenches are then completed.

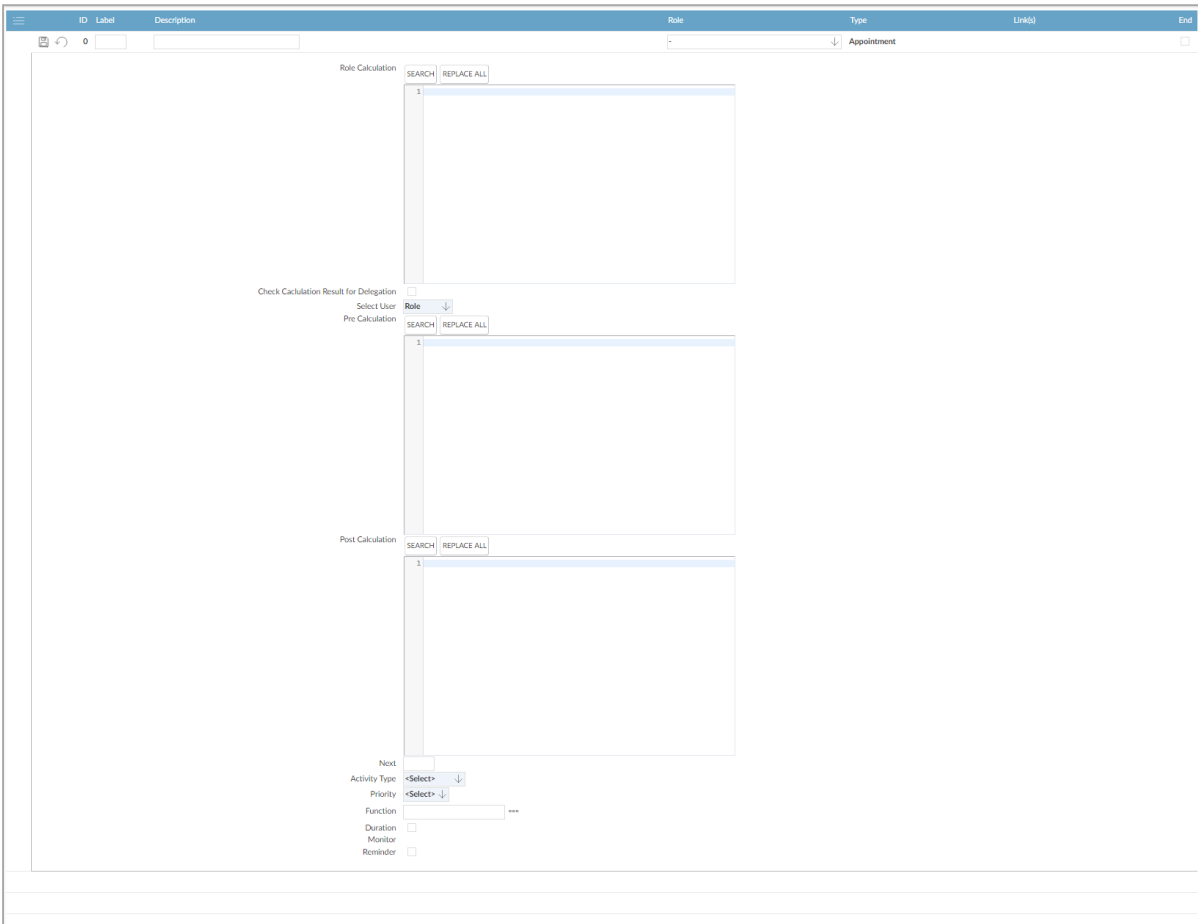
An individual being assigned a task from any of the options above must be a valid COINS User and must be licensed for Workflow.



If an Email stage is being used, a valid email address needs to be assigned either against the User ID, or if appropriate, the associated COINS HR Personnel record. If an SMS stage is to be used it is necessary to have an associated HR record so that COINS can access the appropriate mobile telephone number.

### 8.9.2.1 WF Stages - Appointment

An appointment will be created on the Users' Activity Workbench. The workflow will proceed to the next stage immediately.



Field	Description
Role Calculation	A calculation to determine the user (or users) who must action this stage; used as an alternative to the Role. This calculation must result in one or more user names. If the role above is specified, the calculation variable "this" is assigned as the result of the role assignment. This can then be further manipulated by this calculation as required.
Check Calculation Result for Delegation	If a workflow stage uses a role calculation, instead of an assigned Role, delegation will not automatically kick-in - so if they are away the calculated role will always apply. You can now set it to check for delegation by ticking this field

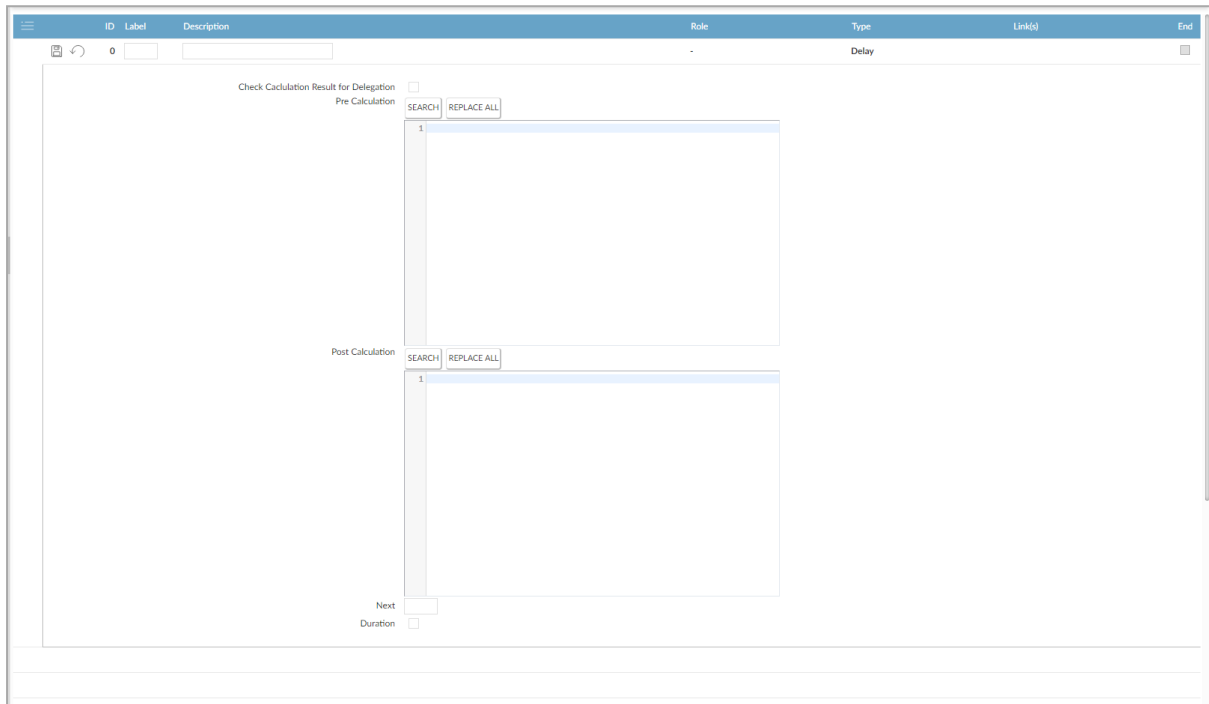


Field	Description
Select User	<p>If the Role or Role Calculation results in a list of users, this determines how COINS treats the list.</p> <p>Role - Whatever the result of the Role or Role Calculation fields is, use that. If the result is two or more users, the task will appear on each user's workbench, but only one of them has to action the task.</p> <p>Individual - If the result is a list, the user who is handing on the task must first select one individual from the list to be the person to action the task.</p> <p>Group - The person who is handing on the task may select an individual, or assign it to everyone in the list.</p>
Pre Calculation	Calculation to be performed before the stage. For example, you could use this to throw an error if a certain condition hasn't been met before the stage is run.
Post Calculation	Calculation to be performed after the stage. For example, you could use this to send an email when the stage has been completed.
Next	The label of another stage to which this stage is linked.
Activity Type	Select the type of appointment or task from the drop down list. The types are user-defined; use Action Types to set them up.
Priority	The priority of the action. You can set an action be either of high, normal or low importance.
Function	The code for the function that this stage links to (plus any function parameters).


Field	Description
Duration	<p>Whether this stage has a duration associated with it. If the stage has duration, the entry on the Activity Workbench will turn red if the stage is overdue. Ticking this will allow the following to be defined:</p> <p>The duration of the stage (in the units shown in the next field).</p> <div data-bbox="635 689 1353 770" style="border: 1px solid #ccc; padding: 5px; margin: 10px 0;"> <p><b>Duration:</b> <input checked="" type="checkbox"/> <input type="text"/> <span style="border: 1px solid #ccc; padding: 2px 5px;">&lt;Select&gt;</span> <span style="border: 1px solid #ccc; padding: 2px 5px;">Relative</span></p> <p><b>Monitor:</b> <span style="border: 1px solid #ccc; padding: 2px 5px;">None</span></p> </div> <p>The units for the duration of the stage.</p> <p>When the duration is calculated from:</p> <p>Absolute = From the start of the workflow.</p> <p>Relative = From the start of this stage.</p>
Monitor	<p>The person whom COINS should notify if the stage is overdue. If the stage is overdue, COINS creates a duplicate action on the monitor's workbench, as an escalation type activity.</p> <p>Associated Parameters - SY/WFESC</p> <p>The activity workbench activity type to create if a workflow is escalated to the workflow monitor.</p> <p>You can configure different activity types for the activity workbench so that these escalation tasks stand out.</p>
Reminder	<p>Whether to generate an Outlook reminder for the task. If you are using Outlook integration, this will generate an Outlook reminder for the user.</p>

### 8.9.2.2 WF Stages - Delay

A time delay can be added to a workflow. Simply define a number, the unit of time and whether the delay should be relative to the stage or absolute to the workflow.

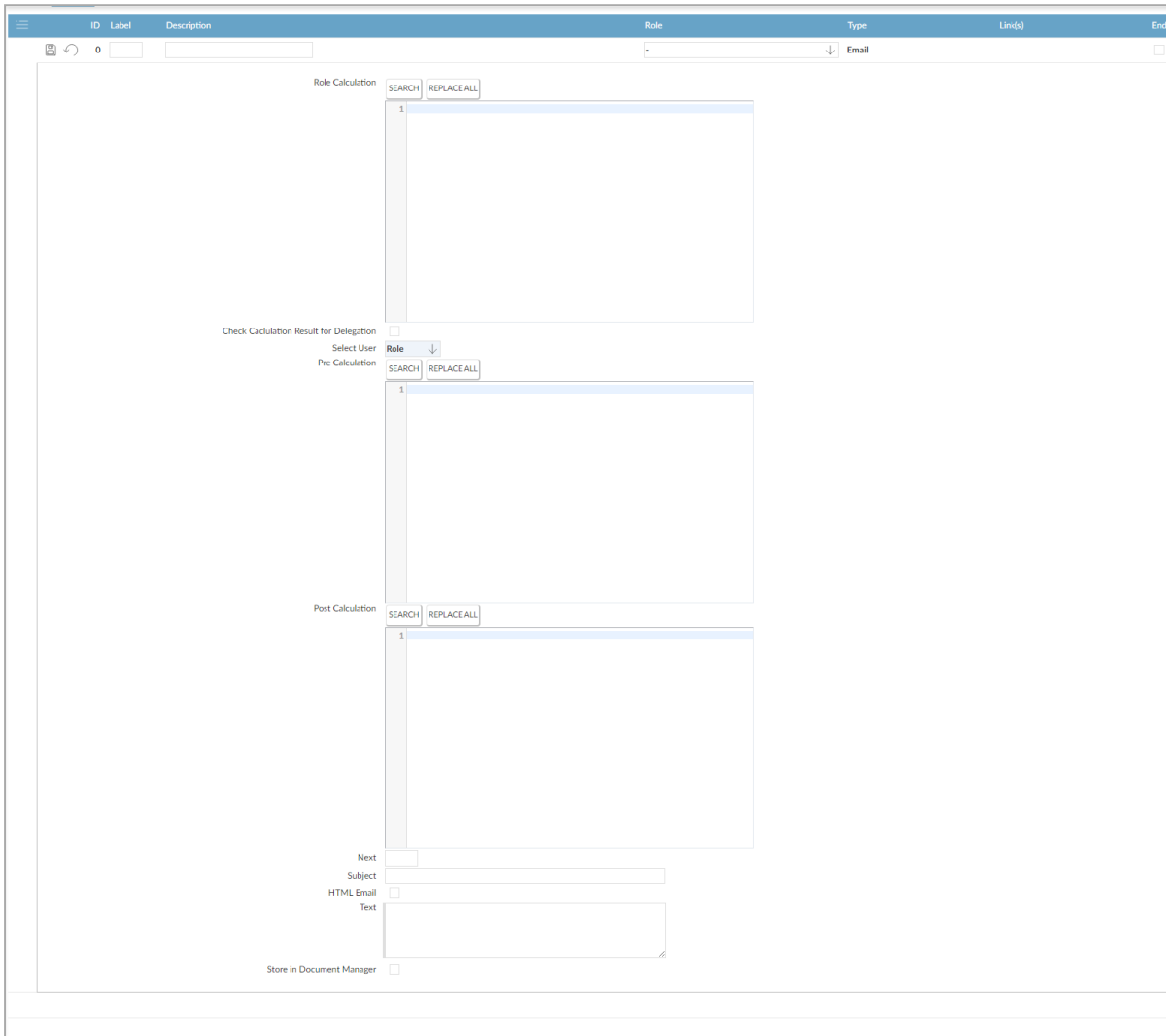


	Description
Check Calculation Result for Delegation	If a workflow stage uses a role calculation, instead of an assigned Role, delegation will not automatically kick-in - so if they are away the calculated role will always apply. You can now set it to check for delegation by ticking this field

	Description
Pre Calculation	Calculation to be performed before the stage. For example, you could use this to throw an error if a certain condition hasn't been met before the stage is run.
Post Calculation	Calculation to be performed after the stage. For example, you could use this to send an email when the stage has been completed.
Next	The label of another stage to which this stage is linked.
Duration	<p>Whether this stage has a duration associated with it. If the stage has a duration, the entry on the Activity Workbench will turn red if the stage is overdue. Ticking this will allow the following to be defined:</p> <p>The duration of the stage (in the units shown in the next field).</p>  <p>When the duration is calculated from:            Absolute = From the start of the workflow.            Relative = From the start of this stage.</p>

### 8.9.2.3 WF Stages - Email

The workflow can send emails based on the role. Define the Subject and Text of the email. Curly braces {} can be used to allow replacement of pertinent details relating to the specific workflow.

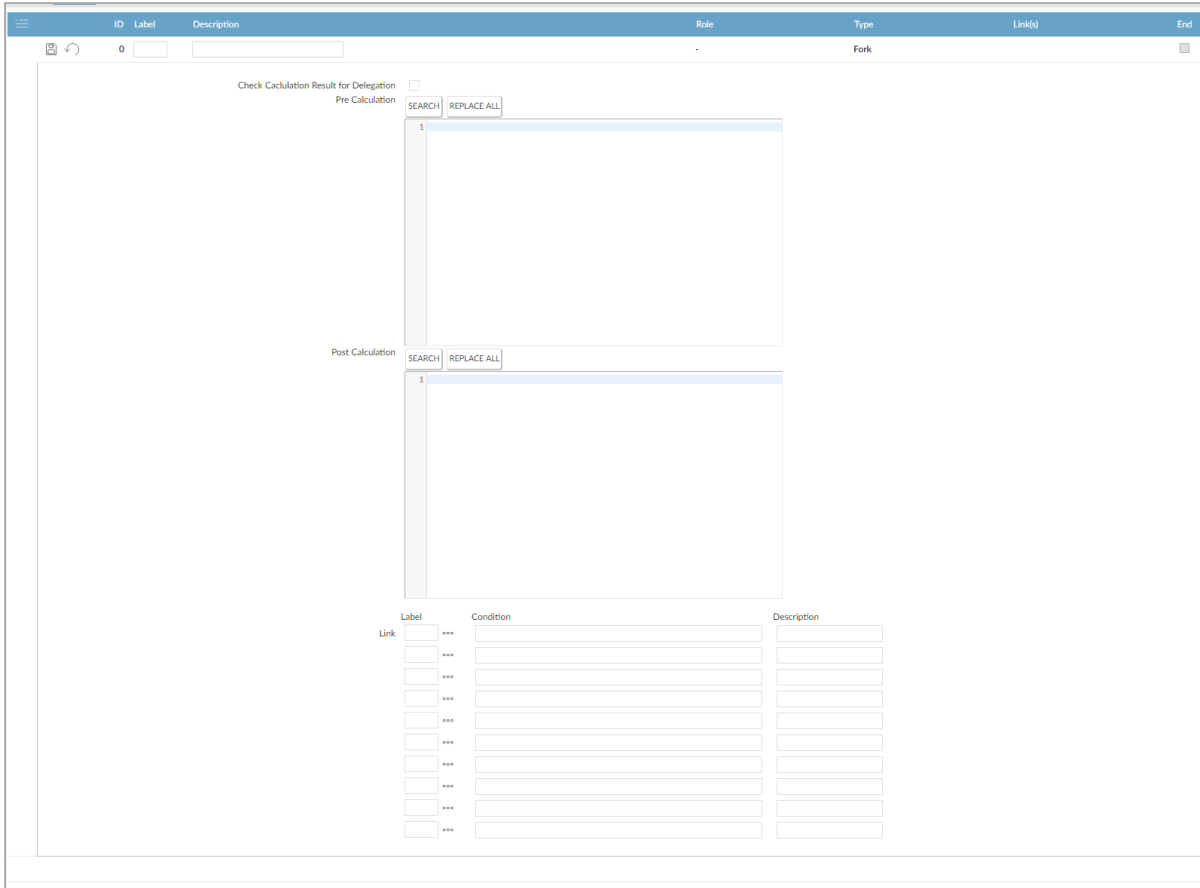


Field	Description
Role Calculation	A calculation to determine the user (or users) who must action this stage; used as an alternative to the Role. This calculation must result in one or more user names. If the role above is specified, the calculation variable “this” is assigned as the result of the role assignment. This can then be further manipulated by this calculation as required.

Field	Description
Check Calculation Result for Delegation	If a workflow stage uses a role calculation, instead of an assigned Role, delegation will not automatically kick-in - so if they are away the calculated role will always apply. You can now set it to check for delegation by ticking this field
Select User	<p>If the Role or Role Calculation results in a list of users, this determines how COINS treats the list.</p> <p>Role - Whatever the result of the Role or Role Calculation fields is, use that. If the result is two or more users, the task will appear on each user's workbench, but only one of them has to action the task.</p> <p>Individual - If the result is a list, the user who is handing on the task must first select one individual from the list to be the person to action the task.</p> <p>Group - The person who is handing on the task may select an individual, or assign it to everyone in the list.</p>
Pre Calculation	Calculation to be performed before the stage. For example, you could use this to throw an error if a certain condition hasn't been met before the stage is run.
Post Calculation	Calculation to be performed after the stage. For example, you could use this to send an email when the stage has been completed.
Next	The label of another stage to which this stage is linked.
Subject	Subject of the email that will be sent for this stage of the workflow.
HTML Email	Select if the email should be formatted with HTML
Text	The text of the email that will be sent for this stage of the workflow.
Store in Document Manager	Whether the message sent should be stored in document manager.

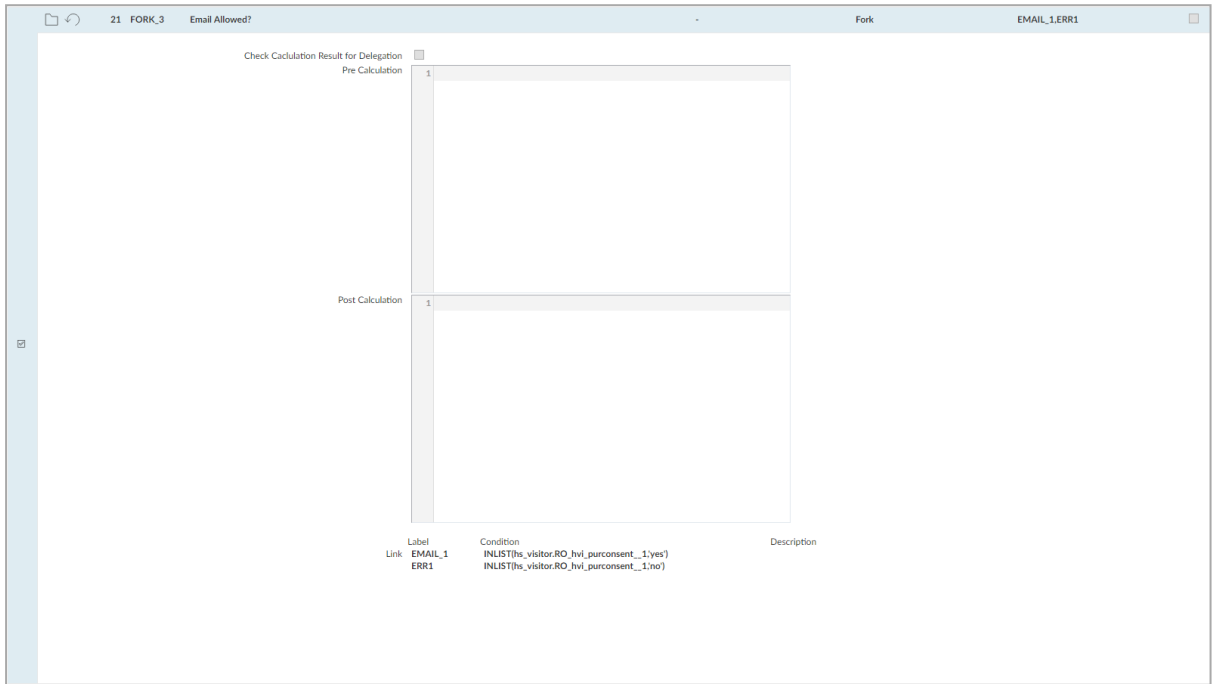
### 8.9.2.4 WF Stages - Fork

A fork is added to create multiple branches in a workflow.



Field	Description
Check Calculation Result for Delegation	If a workflow stage uses a role calculation, instead of an assigned Role, delegation will not automatically kick-in - so if they are away the calculated role will always apply. You can now set it to check for delegation by ticking this field
Pre Calculation	Calculation to be performed before the stage. For example, you could use this to throw an error if a certain condition hasn't been met before the stage is run.
Post Calculation	Calculation to be performed after the stage. For example, you could use this to send an email when the stage has been completed.
Link	The label of another stage to which this stage is linked.
Condition	A condition that must be true to enable the link. If this calculation evaluates to zero the link on this page is not enabled. If the calculation evaluates to a non-zero, or is blank, the link is enabled.

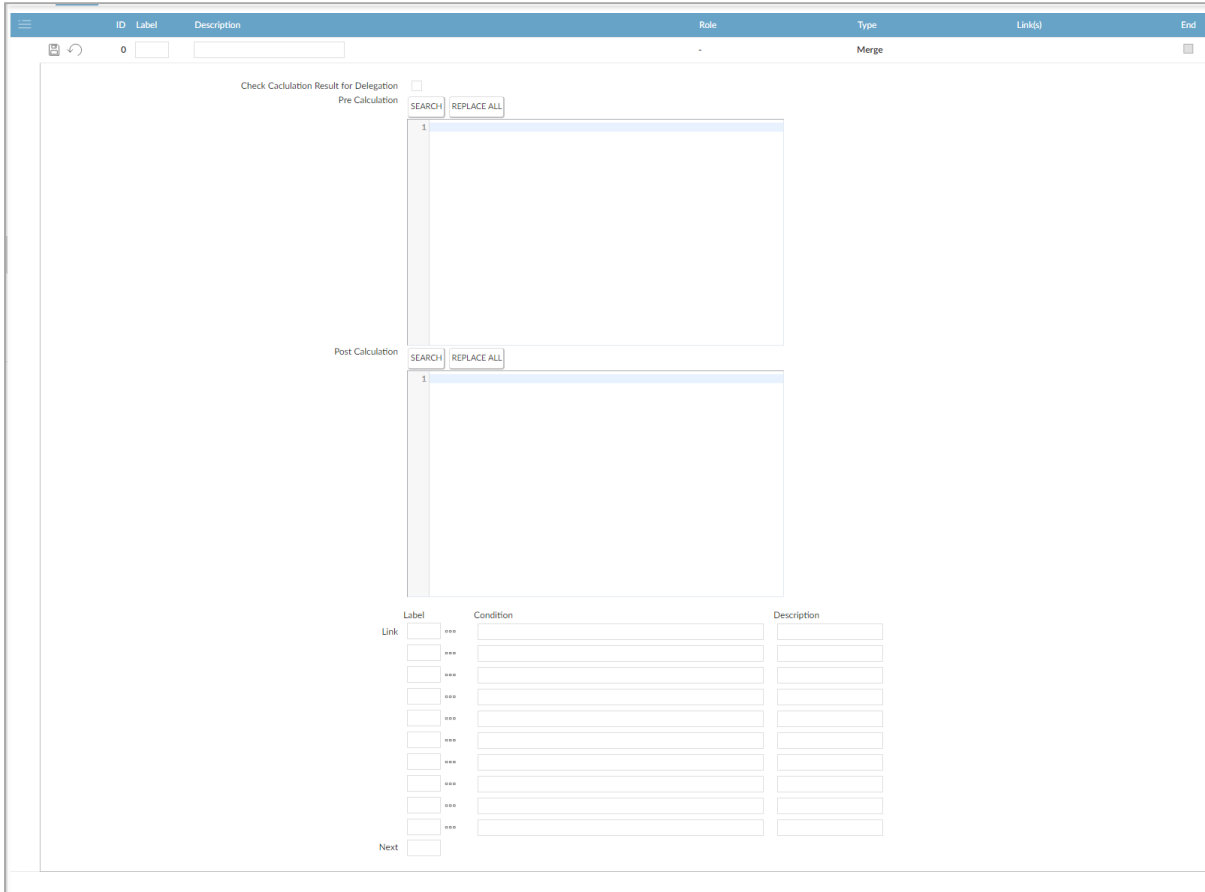
In this example a fork is defined based on the results of a test against the values of a table field.





### 8.9.2.5 WF Stages - Merge

A merge stage is used to bring two or more forks of a workflow back together. The merge stage will not be completed until all stages linked to it are complete.



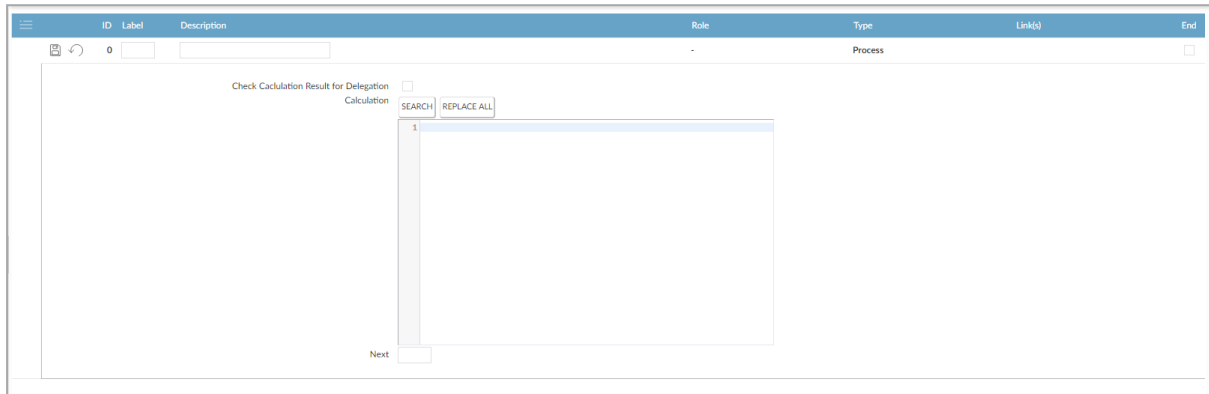
Field	Description
Check Calculation Result for Delegation	If a workflow stage uses a role calculation, instead of an assigned Role, delegation will not automatically kick-in - so if they are away the calculated role will always apply. You can now set it to check for delegation by ticking this field
Pre Calculation	Calculation to be performed before the stage. For example, you could use this to throw an error if a certain condition hasn't been met before the stage is run.
Post Calculation	Calculation to be performed after the stage. For example, you could use this to send an email when the stage has been completed.
Link	The label of another stage to which this stage is linked.



Field	Description
Condition	A condition that must be true to enable the link. If this calculation evaluates to zero the link on this page is not enabled. If the calculation evaluates to a non-zero, or is blank, the link is enabled.

### 8.9.2.6 WF Stages - Process

A process stage can be used to perform a calculation, usually against the COINS database within the calculation. Processes do not require user interaction.



Field	Description
Check Calculation Result for Delegation	If a workflow stage uses a role calculation, instead of an assigned Role, delegation will not automatically kick-in - so if they are away the calculated role will always apply. You can now set it to check for delegation by ticking this field
Calculation	Calculation to be performed - update a hold code for example (see below)
Next	The label of another stage to which this stage is linked.

**Figure 30: Generating an error message if a condition is not met.**





Figure 31: Approving an Invoice (releasing hold codes)

The screenshot shows a software interface with a table at the top containing columns: ID, Label, Description, Role, Type, Link(s), and End. The table has one row with the following data: ID: 35, Label: INVREL, Description: Release Hold Codes, Role: -, Type: Process, Link(s): --END, End: [icon].

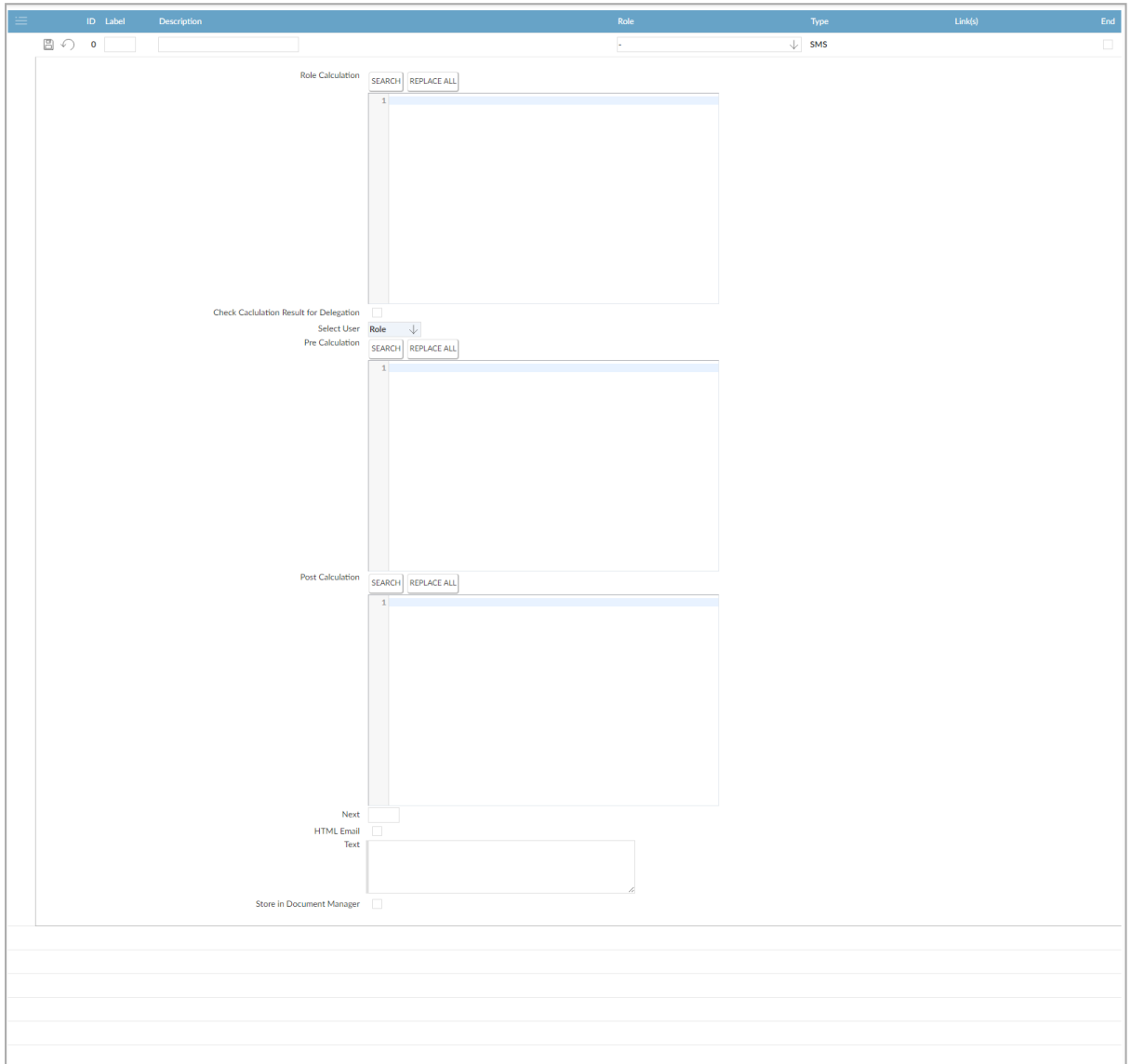
Below the table, there is a section titled "Check Caclulation Result for Delegation" with a checkbox. Underneath is a "Calculation" area with "SEARCH" and "REPLACE ALL" buttons. A text area contains the following code:

```
1 method("ain-rsp.setRowids",op_invoiceRowid);
2 method("ain-rsp.setRSPAction","U");
3 method("ain-rsp.setBufferField","ain_hcode",0,"");
4 method("ain-rsp.setBufferField","ain_hold",0,"N");
5 method("ain-rsp.setBufferField","ain_hcode2",0,"");
6 method("ain-rsp.setBufferField","ain_hold2",0,"N");
7 method("ain-rsp.commitCurrent");
```

At the bottom of the calculation area, there is a "Next" label and the text "END".

### 8.9.2.7 WF Stages - SMS

It is possible to send information from the workflow via SMS (assuming the necessary configuration to allow SMS from COINS has been completed).



Field	Description
Role Calculation	A calculation to determine the user (or users) who must action this stage; used as an alternative to the Role. This calculation must result in one or more user names. If the role above is specified, the calculation variable "this" is assigned as the result of the role assignment. This can then be further manipulated by this calculation as required.



Field	Description
Check Calculation Result for Delegation	If a workflow stage uses a role calculation, instead of an assigned Role, delegation will not automatically kick-in - so if they are away the calculated role will always apply. You can now set it to check for delegation by ticking this field
Select User	<p>If the Role or Role Calculation results in a list of users, this determines how COINS treats the list.</p> <p>Role - Whatever the result of the Role or Role Calculation fields is, use that. If the result is two or more users, the task will appear on each user's workbench, but only one of them has to action the task.</p> <p>Individual - If the result is a list, the user who is handing on the task must first select one individual from the list to be the person to action the task.</p> <p>Group - The person who is handing on the task may select an individual, or assign it to everyone in the list.</p>
Pre Calculation	Calculation to be performed before the stage. For example, you could use this to throw an error if a certain condition hasn't been met before the stage is run.
Post Calculation	Calculation to be performed after the stage. For example, you could use this to send an email when the stage has been completed.
Next	The label of another stage to which this stage is linked.
Text	The text of the SMSI that will be sent for this stage of the workflow.
Store in Document Manager	Whether the message sent should be stored in document manager.

### 8.9.2.8 WF Stages - Stage

A stage is where the workflow will require user interaction to complete the required part of the process and move the workflow onto the next stage.



The screenshot displays the configuration interface for a workflow stage. At the top, there is a table with columns: ID, Label, Description, Role, Type, Link(s), and End. Below this table, there are three sections for calculations: 'Role Calculation', 'Pre Calculation', and 'Post Calculation'. Each section contains a 'SEARCH' button and a 'REPLACE ALL' button. A 'Select User' dropdown menu is located between the 'Role Calculation' and 'Pre Calculation' sections. At the bottom of the interface, there is a table with columns: Label, Condition, and Description. Below this table, there are several configuration options: 'Default Link', 'Activity Type' (with a dropdown menu), 'Priority' (with a dropdown menu), 'Function' (with a text input field), 'Workflow Email' (checkbox), 'HTML Email' (checkbox), 'Text' (with a text area), 'Duration' (checkbox), 'Monitor' (checkbox), 'Reminder' (checkbox), and 'Timeout' (checkbox).

Field	Description
Role Calculation	A calculation to determine the user (or users) who must action this stage; used as an alternative to the Role. This calculation must result in one or more user names. If the role above is specified, the calculation variable “this” is assigned as the result of the role assignment. This can then be further manipulated by this calculation as required.
Check Calculation Result for Delegation	If a workflow stage uses a role calculation, instead of an assigned Role, delegation will not automatically kick-in - so if they are away the calculated role will always apply. You can now set it to check for delegation by ticking this field
Select User	<p>If the Role or Role Calculation results in a list of users, this determines how COINS treats the list.</p> <p>Role - Whatever the result of the Role or Role Calculation fields is, use that. If the result is two or more users, the task will appear on each user's workbench, but only one of them has to action the task.</p> <p>Individual - If the result is a list, the user who is handing on the task must first select one individual from the list to be the person to action the task.</p> <p>Group - The person who is handing on the task may select an individual, or assign it to everyone in the list.</p>
Pre Calculation	Calculation to be performed before the stage. For example, you could use this to throw an error if a certain condition hasn't been met before the stage is run.
Post Calculation	Calculation to be performed after the stage. For example, you could use this to send an email when the stage has been completed.
Link	The label of another stage to which this stage is linked.
Condition	<p>A condition that must be true to enable the link. If this calculation evaluates to zero the link on this page is not enabled. If the calculation evaluates to a non-zero, or is blank, the link is enabled.</p> <p>If no condition is specified, the user must manually decide which of the specified links is to be enabled.</p>



Field	Description
Default Link	A default link can be specified on a stage of type STAGE (it must be one of the links defined in the 10 links above). If specified then when the task arrives at the stage the default stage specified is preloaded on the activity workbench task. If then user updates the completed flag then the next stage is already selected. They can also use the complete task browse action and the default link will be selected. This allows the most common link to be defaulted and eases the effort required by the user. The default link is shown on the diagram in green and with an asterisk after the link label and can be changed by double clicking the link and selecting the default link checkbox
Activity Type	The activity type will be the type of action created on the Activity Workbench (determined by local configuration).
Priority	Priority will be high, normal or low.
Function	The function should be a valid COINS function which will be displayed as a link from the Activity Workbench. The function can be left blank if the action is to be performed outside of COINS or a suitable link is not available. The user will manually need to find and perform the task and then complete the stage in COINS.
Workflow Email	Whether an email should be sent to the user of this stage to complete the action. This will only occur if the user has also been configured to request emails for this workflow.
Text	The text of the email that will be sent for this stage of the workflow.

Field	Description
Duration	<p>Whether this stage has a duration associated with it. If the stage has a duration, the entry on the Activity Workbench will turn red if the stage is overdue. Ticking this will allow the following to be defined:</p> <p>The duration of the stage (in the units shown in the next field).</p> <div data-bbox="639 680 1362 763" style="border: 1px solid #ccc; padding: 5px; margin: 10px 0;"> <p><b>Duration:</b> <input checked="" type="checkbox"/> <input type="text"/> &lt;Select&gt; Relative</p> <p><b>Monitor:</b> None</p> </div> <p>The units for the duration of the stage.</p> <p>When the duration is calculated from:</p> <p>Absolute = From the start of the workflow.</p> <p>Relative = From the start of this stage.</p>
Monitor	<p>The person whom COINS should notify if the stage is overdue. If the stage is overdue, COINS creates a duplicate action on the monitor's workbench, as an escalation type activity.</p> <p>Associated Parameters - SY/WFESC</p> <p>The activity workbench activity type to create if a workflow is escalated to the workflow monitor. You can configure different activity types for the activity workbench so that these escalation tasks stand out.</p>
Reminder	<p>Whether to generate an Outlook reminder for the task. If you are using Outlook integration, this will generate an Outlook reminder for the user.</p>



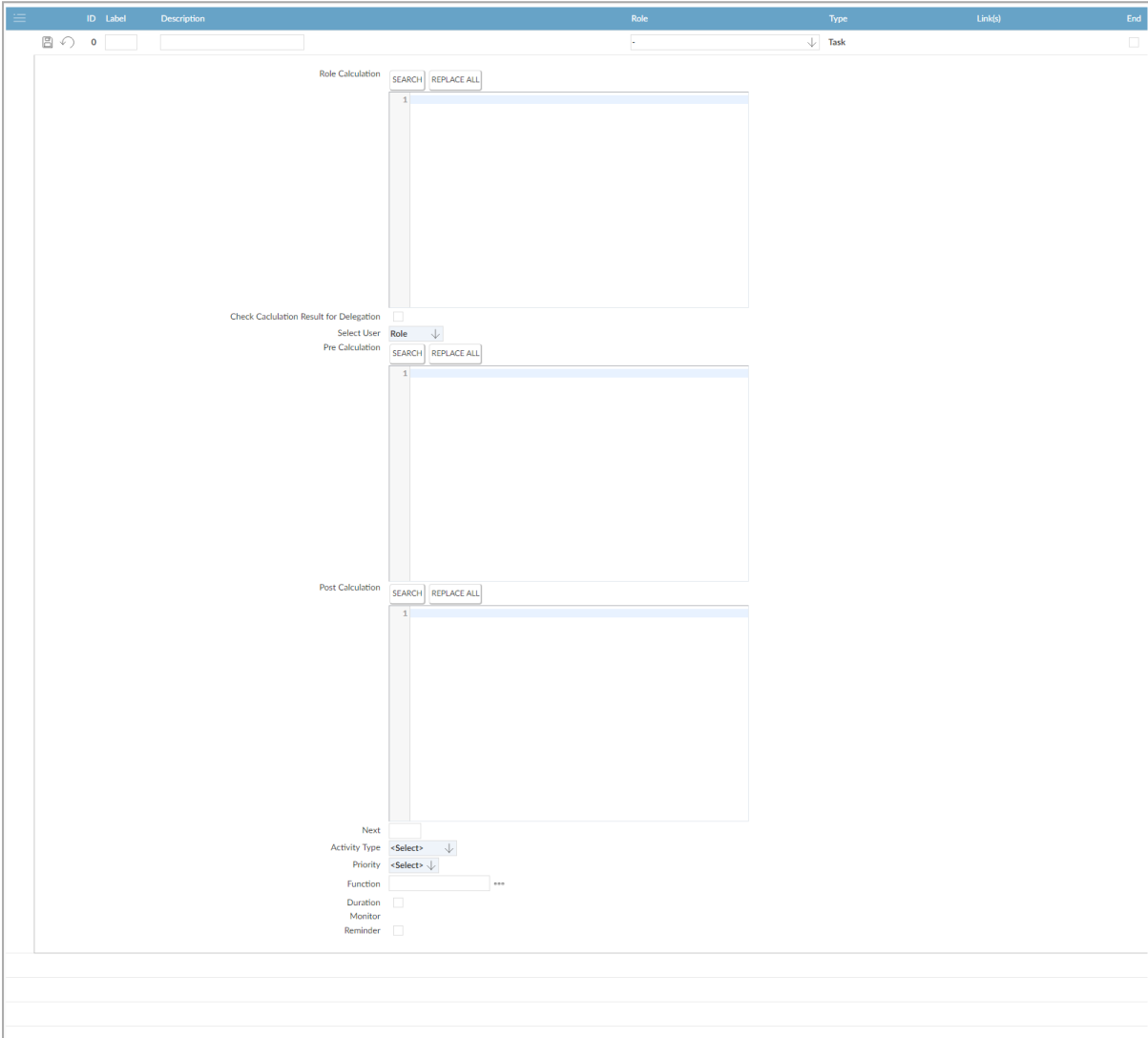
Field	Description
Timeout	<p>If ticked the following will be displayed:</p> <div style="border: 1px solid #ccc; padding: 5px; margin: 10px 0;"> <p><b>Timeout:</b> <input checked="" type="checkbox"/> <input type="text"/> <input type="text" value="&lt;Select&gt;"/></p> <p><b>Timeout Stage:</b> <input type="text"/></p> </div> <p>The default stage label to move to if the timeout is exceeded.</p> <p>If left blank then the next sequential stage will be used.</p>

Below is an example of a stage in PL Invoice Approval where, once the invoice has been reviewed by one role, the next stage available will be determined by the value of the invoice on the invoice.



### 8.9.2.9 WF Stages - Task

A task is similar to a Stage except that the workflow will not wait for a task to be completed before moving onto the next stage – it will simply assign the task to the appropriate role and move to the next linked stage.



Field	Description
Role Calculation	A calculation to determine the user (or users) who must action this stage; used as an alternative to the Role. This calculation must result in one or more user names. If the role above is specified, the calculation variable “this” is assigned as the result of the role assignment. This can then be further manipulated by this calculation as required.

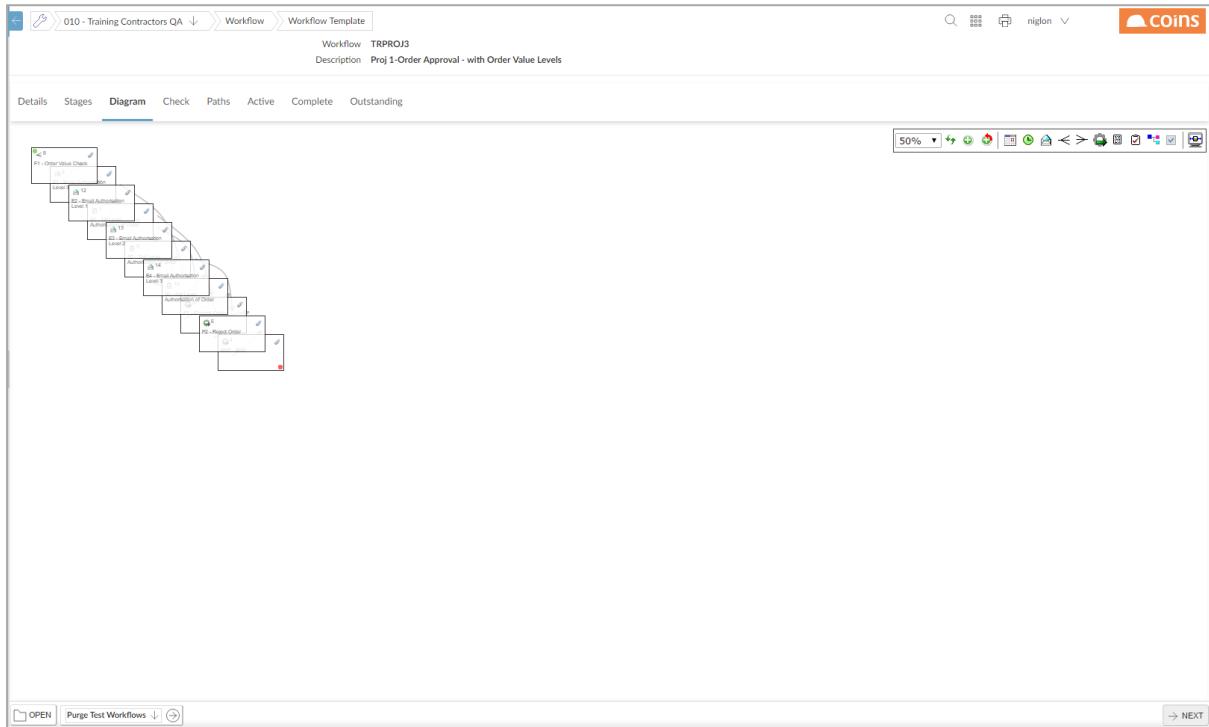
Field	Description
Check Calculation Result for Delegation	If a workflow stage uses a role calculation, instead of an assigned Role, delegation will not automatically kick-in - so if they are away the calculated role will always apply. You can now set it to check for delegation by ticking this field
Select User	<p>If the Role or Role Calculation results in a list of users, this determines how COINS treats the list.</p> <p>Role - Whatever the result of the Role or Role Calculation fields is, use that. If the result is two or more users, the task will appear on each user's workbench, but only one of them has to action the task.</p> <p>Individual - If the result is a list, the user who is handing on the task must first select one individual from the list to be the person to action the task.</p> <p>Group - The person who is handing on the task may select an individual, or assign it to everyone in the list.</p>
Pre Calculation	Calculation to be performed before the stage. For example, you could use this to throw an error if a certain condition hasn't been met before the stage is run.
Post Calculation	Calculation to be performed after the stage. For example, you could use this to send an email when the stage has been completed.
Next	The label of another stage to which this stage is linked.
Activity Type	The activity type will be the type of action created on the Activity Workbench (determined by local configuration).
Priority	Priority will be high, normal or low.
Function	The function should be a valid COINS function which will be displayed as a link from the Activity Workbench. The function can be left blank if the action is to be performed outside of COINS or a suitable link is not available. The user will manually need to find and perform the task and then complete the stage in COINS.

Field	Description
Duration	<p>Whether this stage has a duration associated with it. If the stage has a duration, the entry on the Activity Workbench will turn red if the stage is overdue. Ticking this will allow the following to be defined: The duration of the stage (in the units shown in the next field).</p> <div data-bbox="644 562 1366 645" style="border: 1px solid gray; padding: 5px; margin: 10px 0;"> <p><b>Duration:</b> <input checked="" type="checkbox"/> <input type="text"/> <span style="border: 1px solid gray; padding: 2px;">&lt;Select&gt;</span> <span style="border: 1px solid gray; padding: 2px;">Relative</span></p> <p><b>Monitor:</b> <span style="border: 1px solid gray; padding: 2px;">None</span></p> </div> <p>The units for the duration of the stage.</p> <p>When the duration is calculated from:</p> <p>Absolute = From the start of the workflow. Relative = From the start of this stage.</p>
Monitor	<p>The person whom COINS should notify if the stage is overdue. If the stage is overdue, COINS creates a duplicate action on the monitor's workbench, as an escalation type activity.</p> <p>Associated Parameters - SY/WFESC</p> <p>The activity workbench activity type to create if a workflow is escalated to the workflow monitor. You can configure different activity types for the activity workbench so that these escalation tasks stand out.</p>
Reminder	<p>Whether to generate an Outlook reminder for the task. If you are using Outlook integration, this will generate an Outlook reminder for the user.</p>

### 8.9.3 WF Template - Diagram Tab

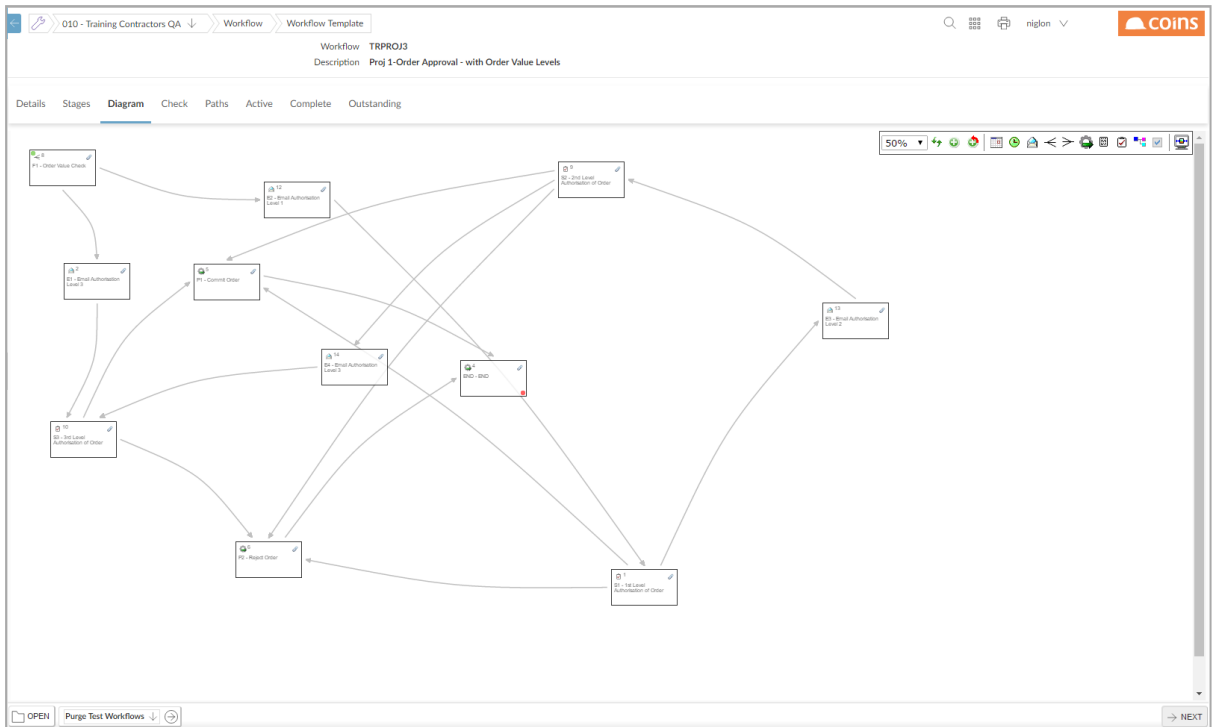
The diagram tab shows and allows maintenance of the workflow diagram.

If stages already exist then the stages will be spaced out linearly on the page.

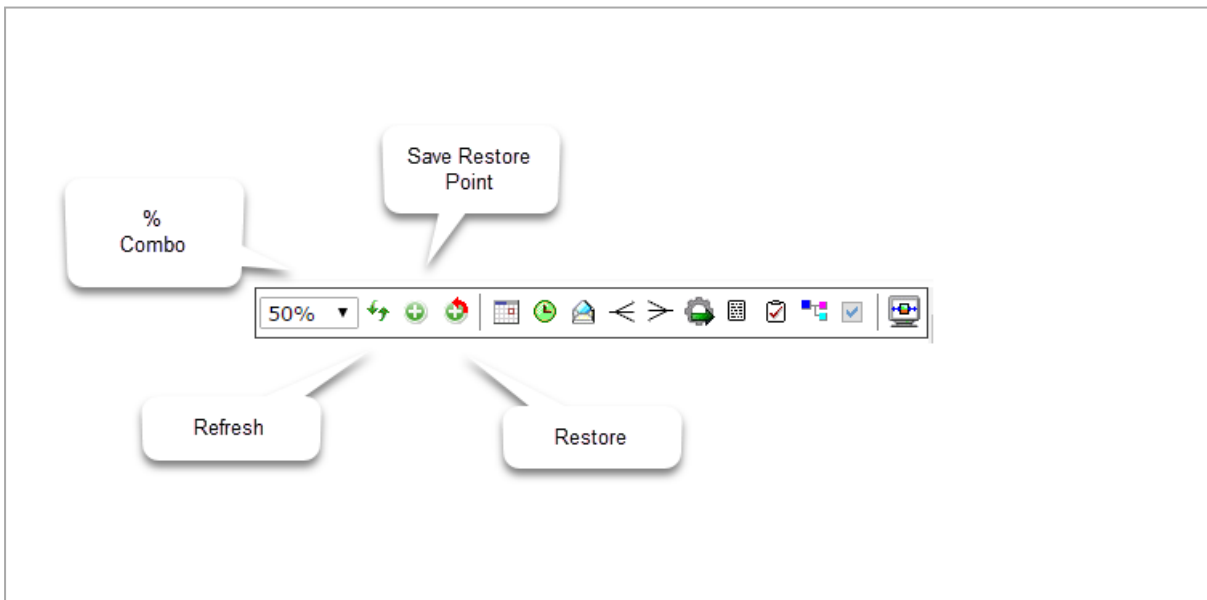
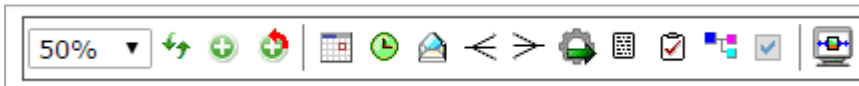


The stages can then be dragged to a desired location on the page. The links between the stages are shown and will move with the stages. Each time the stage is “dropped” the location is saved back to the server. When you next visit the diagram tab the stages will be placed in the position that they were last left.





The toolbar controls the diagram.

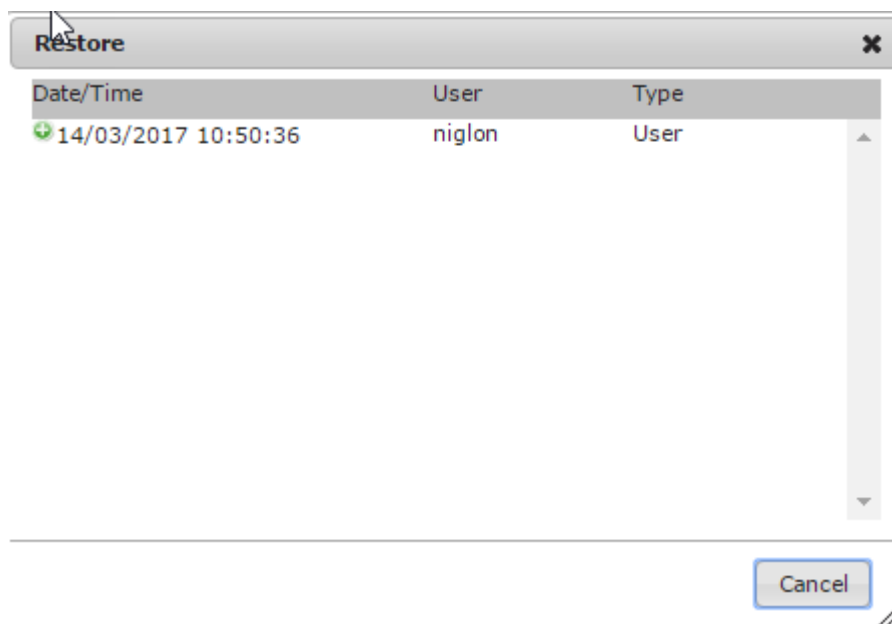


The % combo controls the zoom level and is retained for the page (similar to browse filters). If a new % is selected the diagram is redrawn at that scale.

The refresh button reloads the page. This refreshes all the data from the server and rebuilds and redraws the diagram.

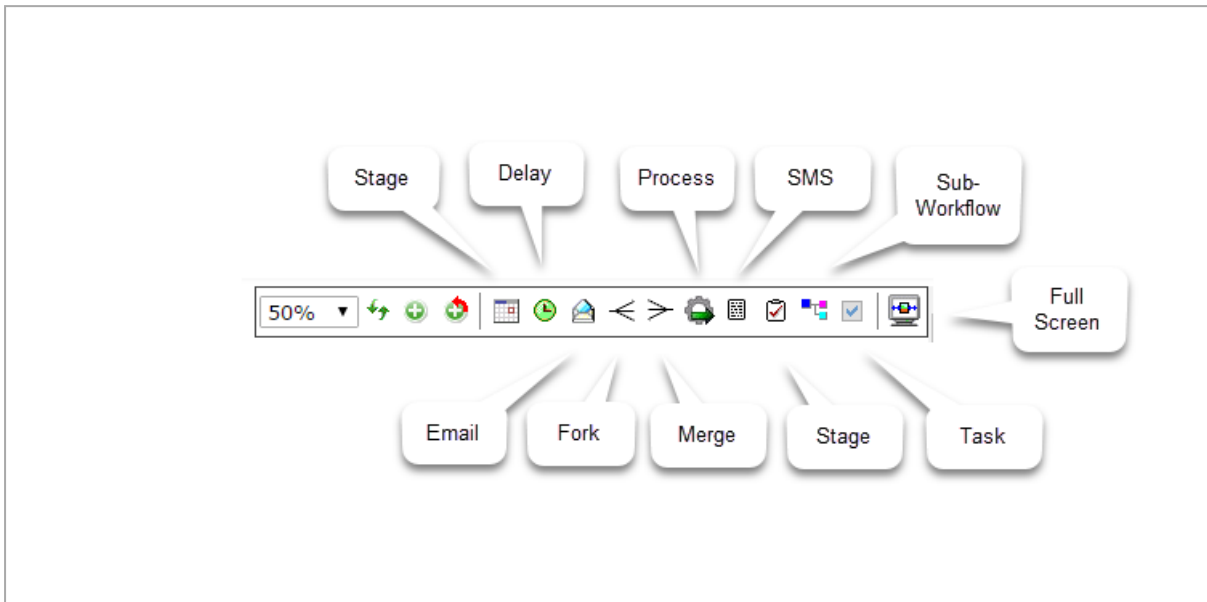
The save restore point button ( takes a snapshot of the workflow template and saves a copy that can be re-instated if required. This allows the user to save the state of the template. 50 back versions will be retained.

The restore button ( allows the user to select one of the 50 restore points and to re-instate as the current version.



Select the plus icon next to the restore point. Selecting a restore point to restore will cause a new restore to be taken (Type=System – just in case you need to get back to the previous position) and then the template and stages are restored.

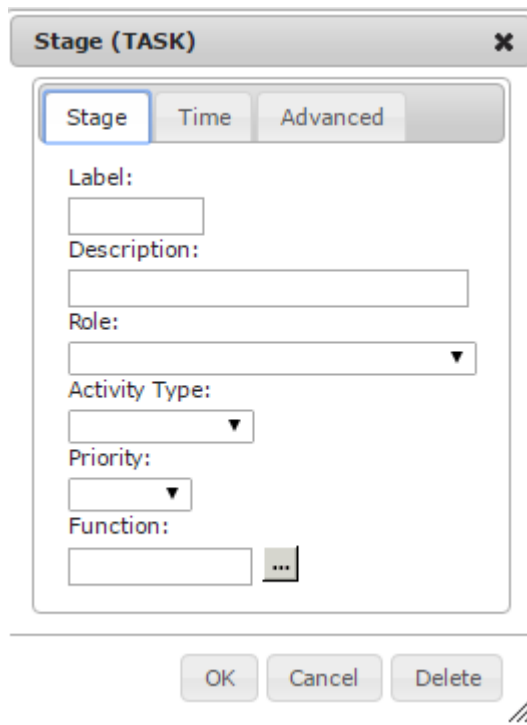
This restores the details on the workflow header e.g. initialization calculations, document, etc.



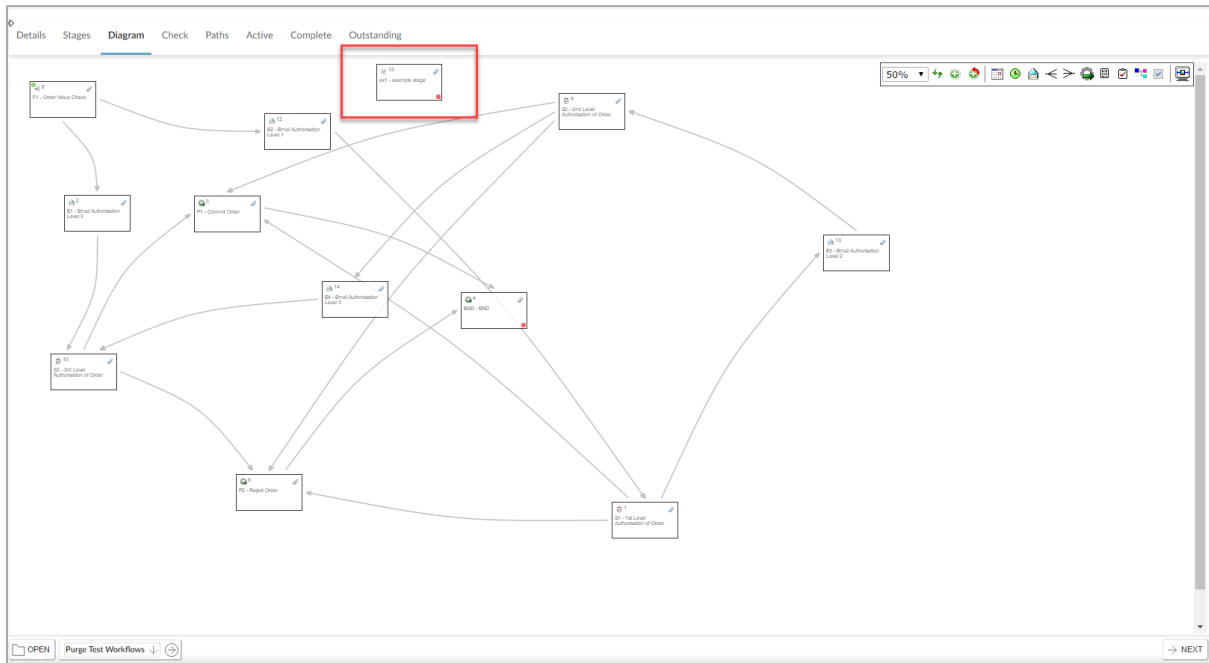
The full screen button will show the diagram in a new browser window which can be maximized to give the full amount of space for editing.

The remaining buttons on the toolbar are for adding new stages of each of the types Appointment, Delay, Fork, Merge, Process, SMS, Stage, Sub Workflow, and Task.

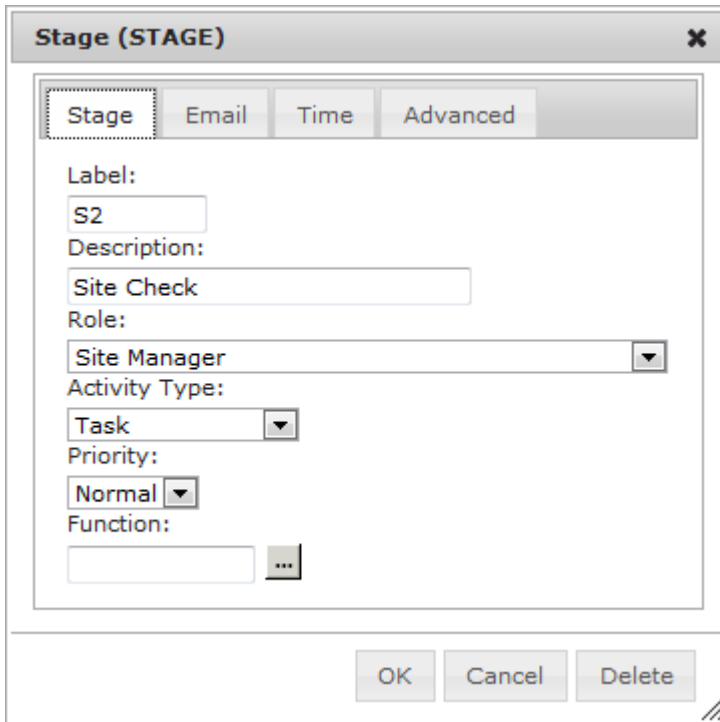
They will show an appropriate dialog to add the stage details.



If you OK the dialog (and there are no errors) then a stage will be created just under the toolbar for you drag on to the diagram canvas.



You can double click a stage to update the details on that stage.



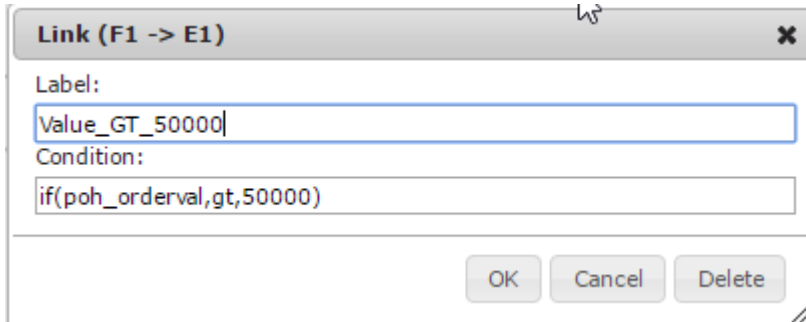
The 'Stage (STAGE)' dialog box is shown with the following configuration:

- Tab: Stage
- Label: S2
- Description: Site Check
- Role: Site Manager
- Activity Type: Task
- Priority: Normal
- Function: [Empty field with a menu icon]

Buttons: OK, Cancel, Delete

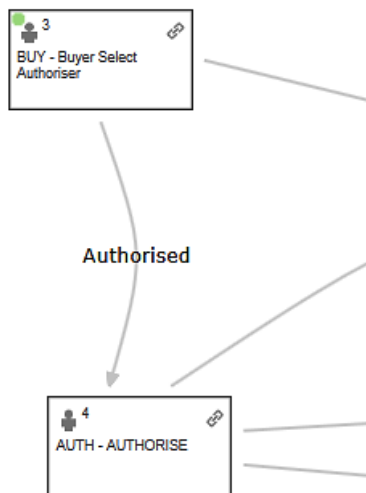
Deleting a stage will cause it to be deleted from the diagram and any links to and from it automatically removed.

You can double click a stage link and a link dialog will be displayed



The image shows a dialog box titled "Link (F1 -> E1)". It has a close button (X) in the top right corner. Below the title, there are two text input fields. The first is labeled "Label:" and contains the text "Value\_GT\_50000". The second is labeled "Condition:" and contains the text "if(poh\_orderval,gt,50000)". At the bottom of the dialog, there are three buttons: "OK", "Cancel", and "Delete".

The label is shown by the link on the diagram.

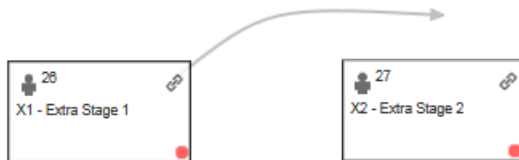


Selecting the delete option will remove the link between the two stages.

You can double click a next link and the next dialog will be shown. This allows you to remove the link by selecting the delete option.



You can drag the cursor from the square in the stage to another stage to create a link between the stages.



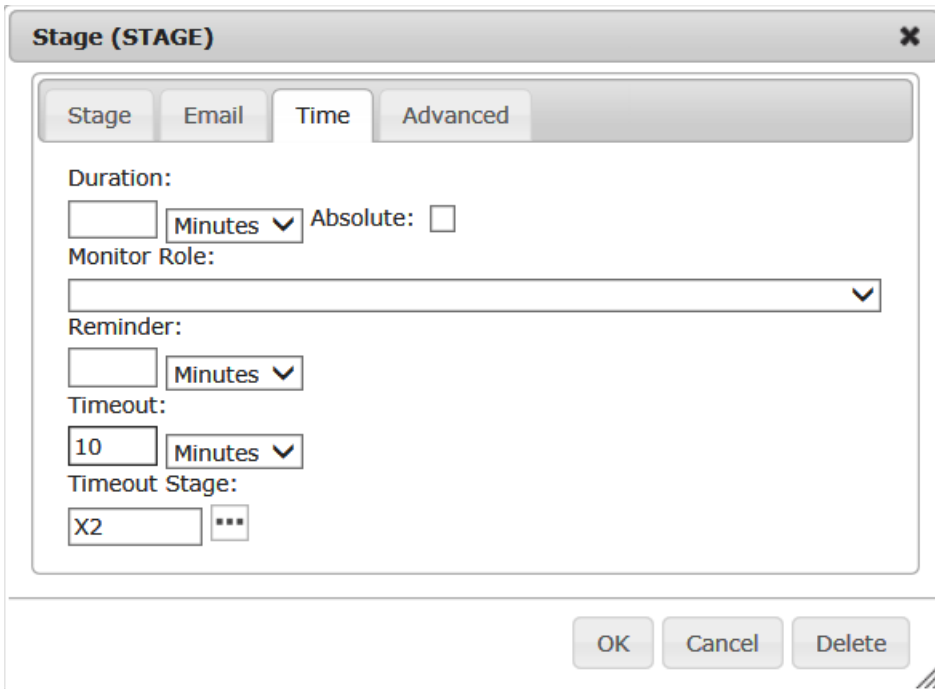
As you have over valid drop zones they will be highlighted



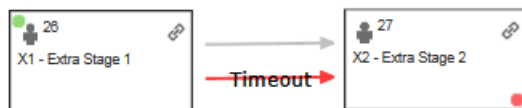
Dropping the link on the stage causes the stage to be created

All the changes described above are written back to the workflow template as each change is made. There is no need to save the template. The restore point feature allows you to restore to specific stable versions if required.

A timeout link can be specified on stages on the Time tab by referring to the label of the target stage.



They are then shown as a red arrow on the diagram and labelled Timeout.



They cannot be deleted by double clicking but only by blanking out the timeout stage on the Time tab.

The meaning and use of the fields for each of the stage types is the same as previously used on the Stages tab. The stage link labels are new and are simply a description of the link to be shown on the diagram.

Any stage that is left without a stage link or next link is automatically set as an End stage. End stages are shown with a red flash in the bottom right hand corner.

The first stage added on the diagram is automatically set as the start stage and shown with a red flash in the top left hand corner. If you wish to change the start stage then you can either re-sequence the stage on the Stages tab so that it is the first stage sequentially or select the Start checkbox on the Advanced tab. This will automatically switch the stage to the first in sequence.

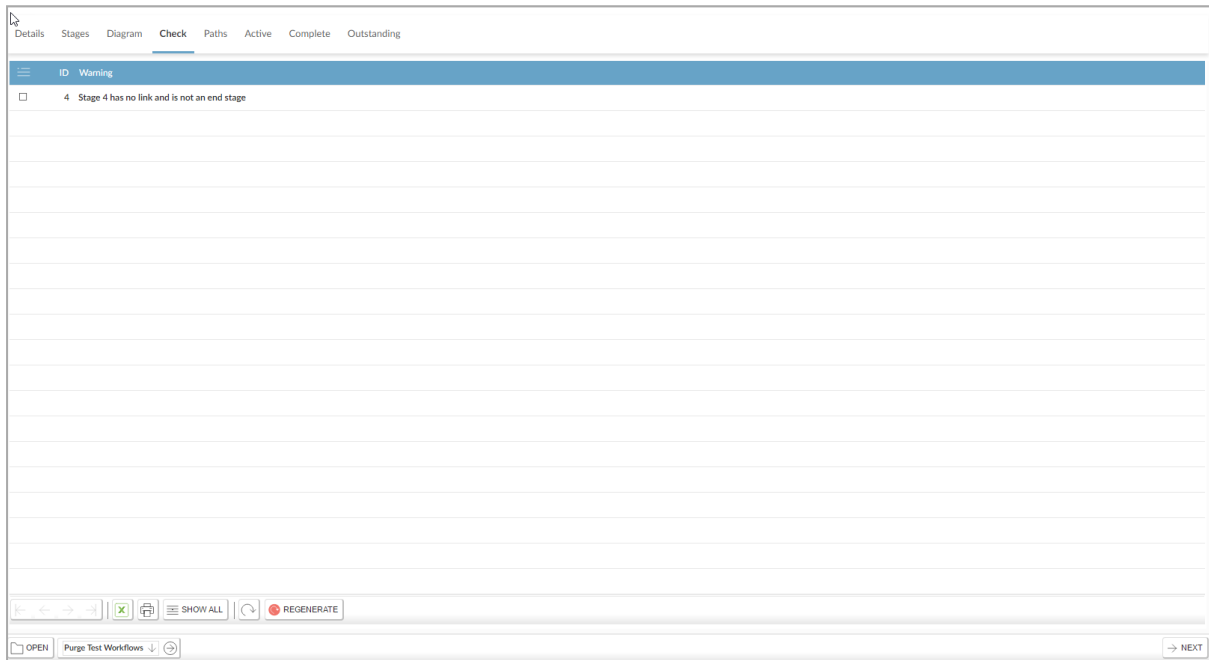


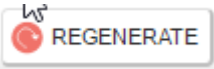
If you switch tabs you may need to refresh the tab data from the server to take in to account changes made on other tabs.



### 8.9.4 WF Template - Check Tab

This tab will provide a quick review of some specific problems which will help to debug the workflow. In the example shown below it is informing us that a stage has no links to other stages but has not been defined as an end stage. These types of errors causes the workflow to fail.

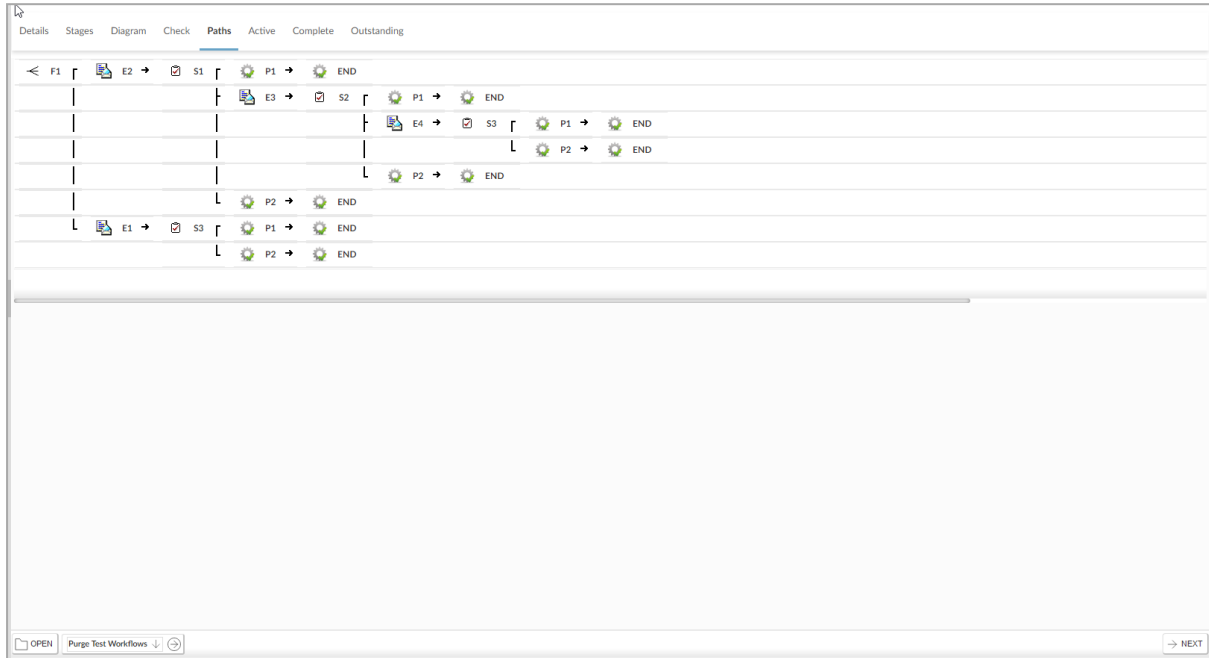


Once the errors have been addressed, regenerate the information on this tab by clicking 

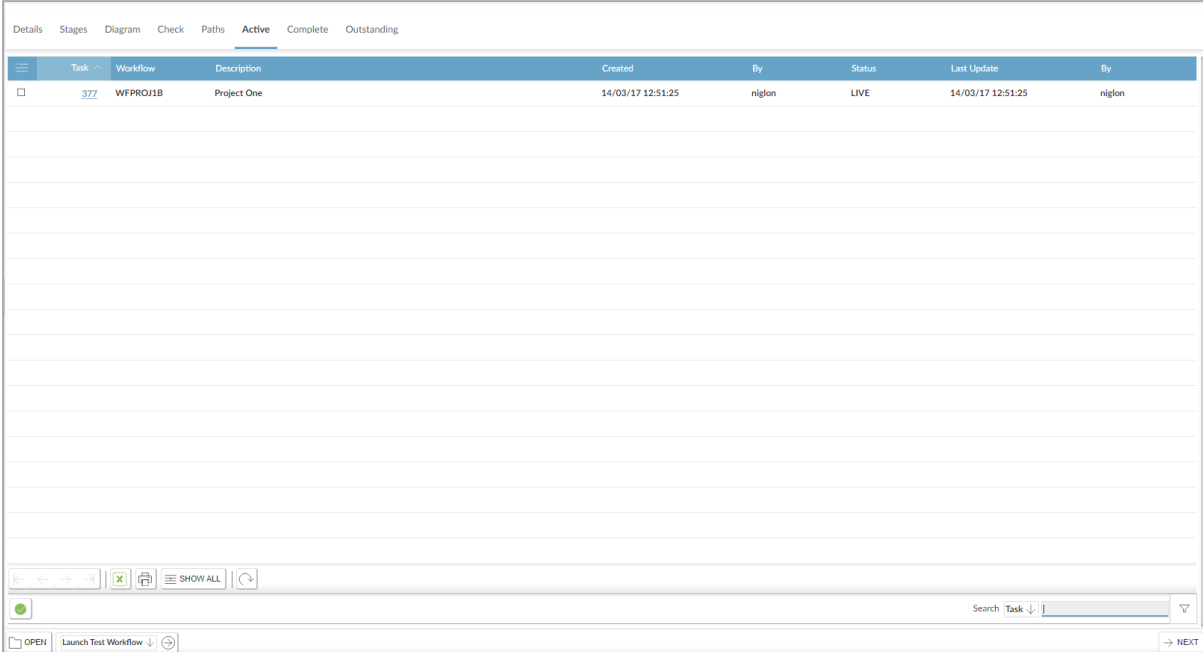
The Id has been added to the stages tab to allow easy identification as to where the errors have been identified.

## 8.9.5 WF Template - Paths Tab

This tab depicts a pictorial view of the workflow. For example:



## 8.9.6 WF Template - Active Tab

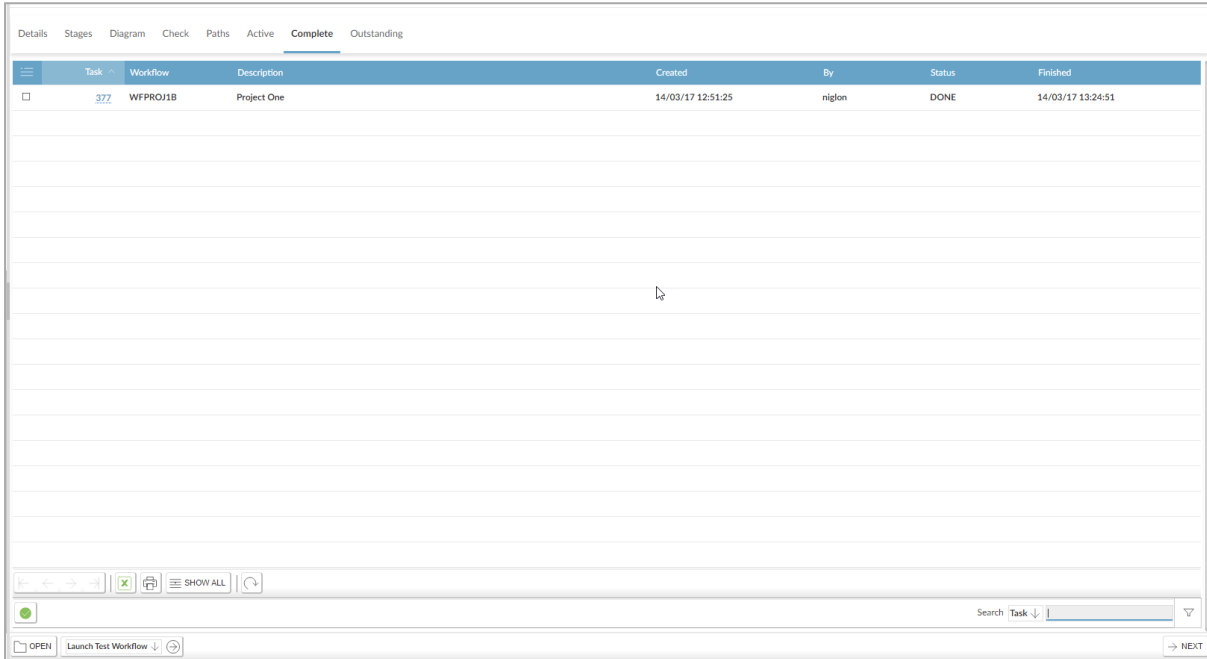


Task	Workflow	Description	Created	By	Status	Last Update	By
377	WFPROJ1B	Project One	14/03/17 12:51:25	niglon	LIVE	14/03/17 12:51:25	niglon

Field	Description
Task	The internal task number of the workflow instance. This field links to the Workflow Summary
Workflow	Workflow template code
Description	The workflow template description
Created	The date the instance was created
By	Who created the workflow instance
Status	The status of the workflow instance
Last update	The date/time of the workflow instance
By	The COINS userID of the user.

## 8.9.7 WF Template - Complete Tab

Workflow Complete Tasks shows you the inactive (completed) tasks for the workflow template



The screenshot shows a web application interface with a navigation menu at the top: Details, Stages, Diagram, Check, Paths, Active, Complete (selected), and Outstanding. Below the menu is a table with the following columns: Task, Workflow, Description, Created, By, Status, and Finished. The table contains one row of data:

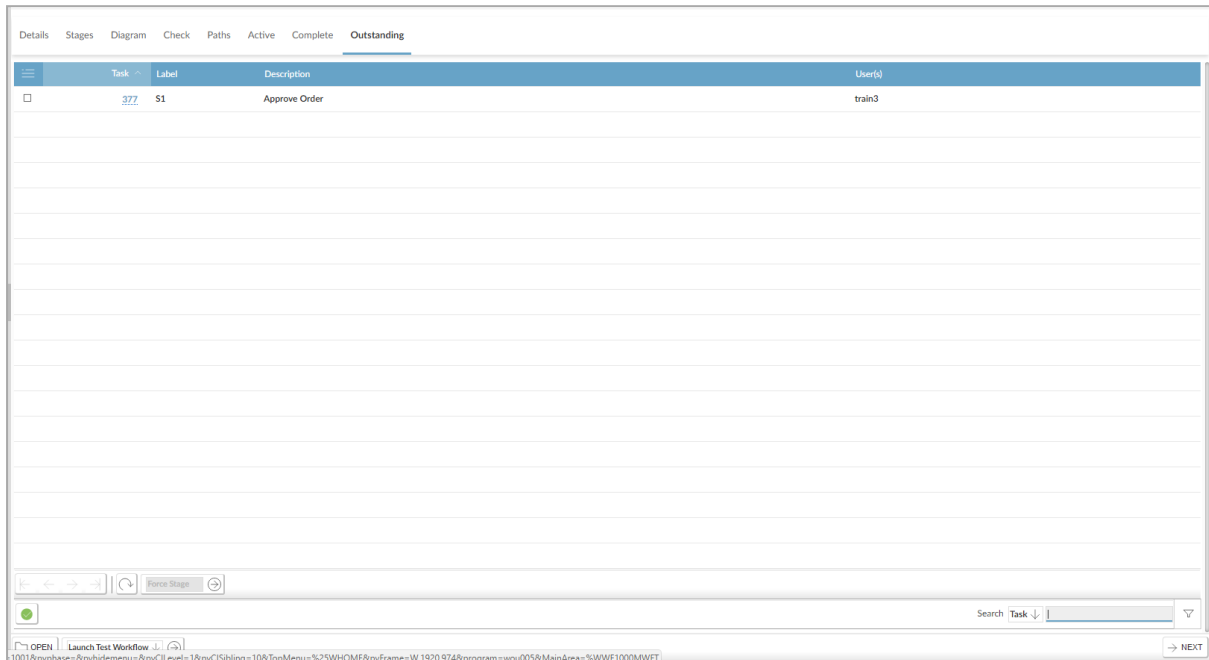
Task	Workflow	Description	Created	By	Status	Finished
377	WFPROJ1B	Project One	14/03/17 12:51:25	niglon	DONE	14/03/17 13:24:51

At the bottom of the table, there are several controls: a search bar with the text 'Search Task', a 'SHOW ALL' button, and a 'NEXT' button.

Field	Description
Task	The internal task number of the workflow instance. This field links to the Workflow Summary
Workflow	Workflow template code
Description	The workflow template description
Created	The date the instance was created
By	Who created the workflow instance
Status	The status of the workflow instance
Finished	The date/time the workflow instance completed

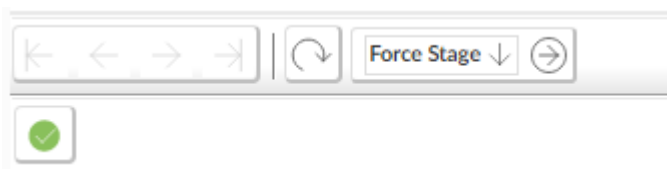
### 8.9.8 WF Template - Outstanding Tab

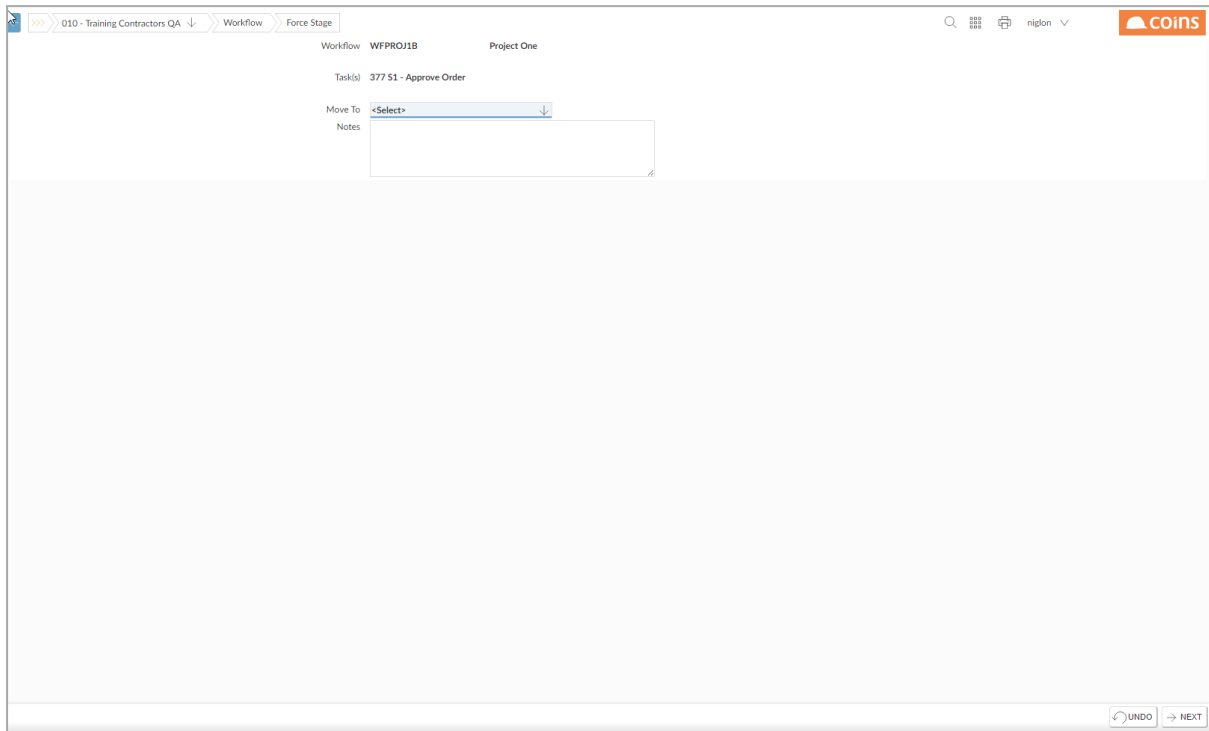
The “Outstanding” shows all outstanding actions on all stages for a workflow.



A link is shown to take you to the task summary (in a new window).

If you select an entry and navigate to the bottom of the screen there is an action that allows you to force the stage to complete and move to another in the workflow.



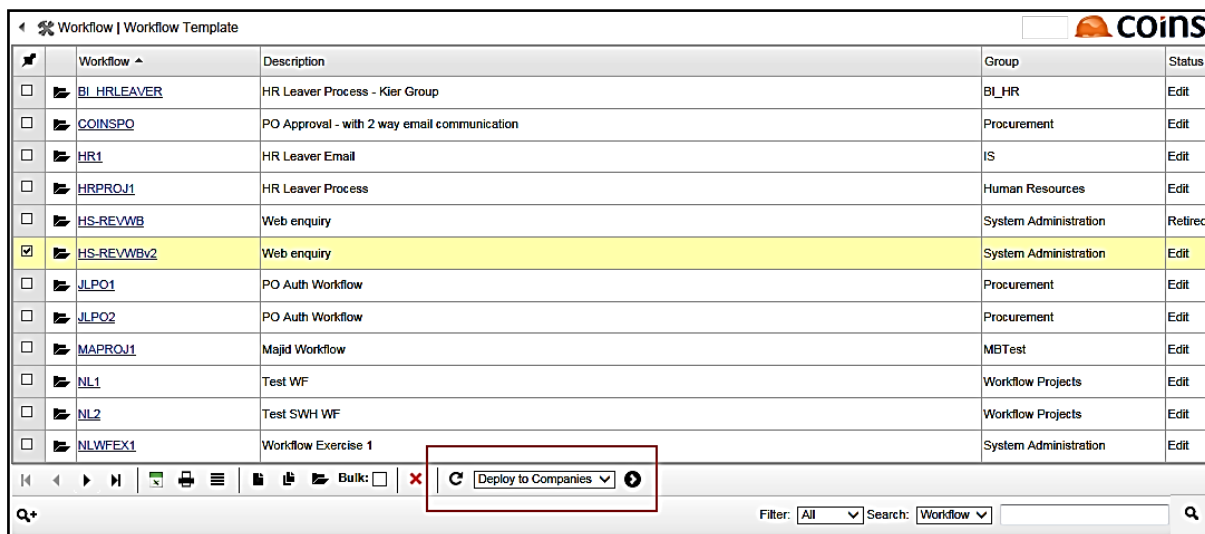


This typically will be used to complete a workflow when events have moved on and the task is no longer required. In order for this to work effectively, it is good design practice to ensure you have an END stage with no processing associated with it that can be selected when forcing a stage.

## 8.10 Workflow Deployment to Companies

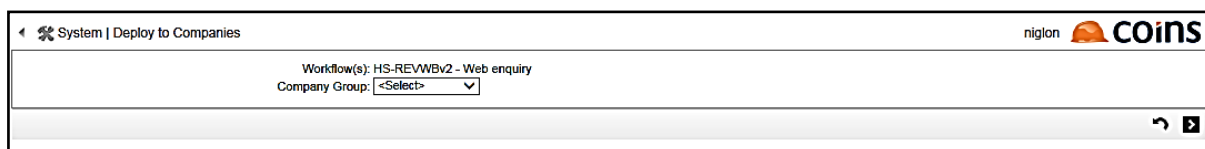
It is possible to copy a workflow template from one company to one or more companies in a single process.

The user can select the workflow or workflows to deploy on the workflow template screen.



Workflow	Description	Group	Status
BI_HRLEAVER	HR Leaver Process - Kier Group	BI_HR	Edit
COINSP0	PO Approval - with 2 way email communication	Procurement	Edit
HR1	HR Leaver Email	IS	Edit
HRPROJ1	HR Leaver Process	Human Resources	Edit
HS-REWB	Web enquiry	System Administration	Retired
HS-REWBv2	Web enquiry	System Administration	Edit
JLPO1	PO Auth Workflow	Procurement	Edit
JLPO2	PO Auth Workflow	Procurement	Edit
MAPROJ1	Majid Workflow	MBTest	Edit
NL1	Test WF	Workflow Projects	Edit
NL2	Test SWH WF	Workflow Projects	Edit
NLWFEX1	Workflow Exercise 1	System Administration	Edit

You can then select the company group (which defines a group of companies) that the template should be copied to.



Workflow(s): HS-REVWBv2 - Web enquiry  
Company Group: <Select>

The system will check that the workflows can be copied to all companies in the company group. The current company (from which the template is being copied) is excluded.

The copy will be prevented (to all companies) if:

- The workflow exists in any company and is status of live
- The workflow exists in any company and there are tasks for it

If all companies in the group pass this criteria then the template from the current company will be copied to all the companies overwriting (delete and create) any versions in the target company that are in status edit.





## 8.11 Workflow Monitor

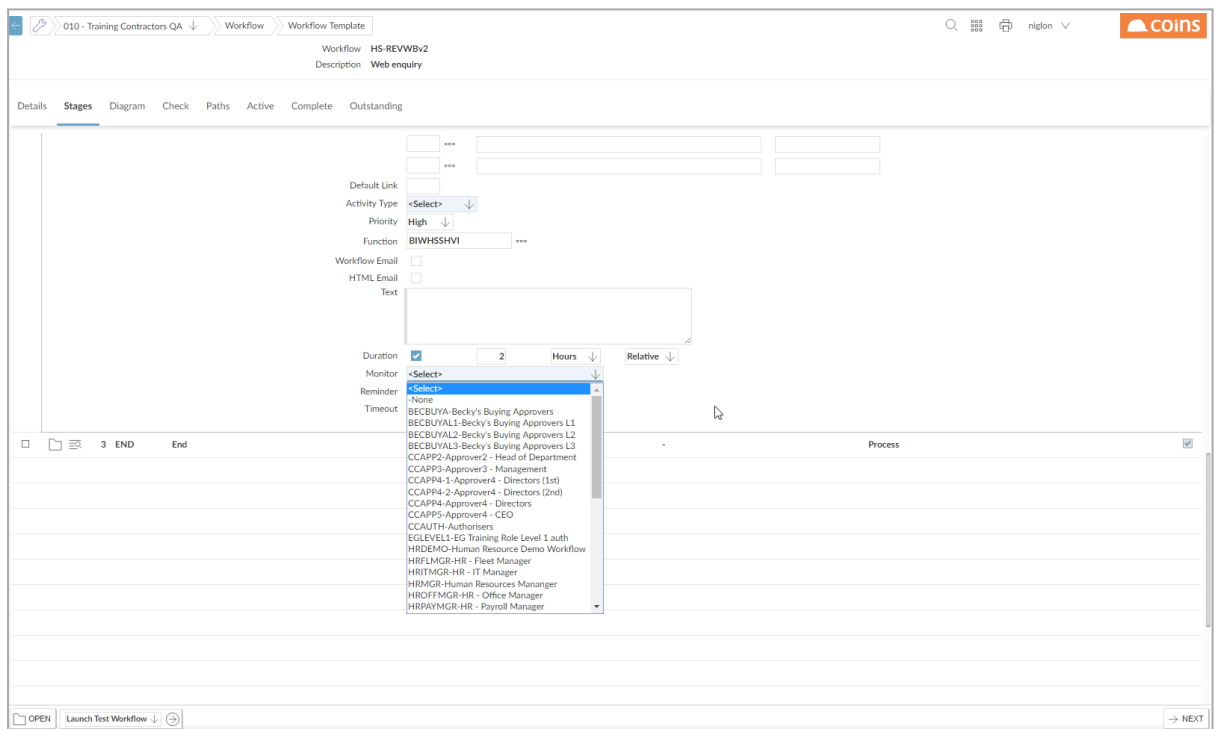
A Monitor is the user who COINS should alert with a duplicate task if a stage extends beyond the expected duration. If a stage is overdue, COINS creates a duplicate action on the monitor's workbench with an escalation type activity.

The escalation type activity is defined in the System Parameter WFESC

NOTE - This MUST be a different action type to that being used within the Workflow stages to allow for appropriate escalation and completion of tasks.

It is possible to assign a Monitor to a workflow as a whole or to individual stages. The monitor specified on the template will be used unless there is a specific monitor specified on the stage itself.

To use the monitor a duration will need to be specified against the stage:



Duration Whether this stage has a duration associated with it. If the stage has a duration, the entry on the Activity Workbench will turn red if the stage is overdue.

The duration of the stage (in the units shown in the next field).

The units for the duration of the stage.

When the duration is calculated from:

Absolute = From the start of the workflow.

Relative = From the start of this stage.

The monitor can then be selected from the list of Roles.

There is a background process (syu9999.p – Background Monitor) which will need to be run against the COINS database so that as Stages become overdue the appropriate monitor can be applied. These processes will also be required by any Delay stages to ensure that once the delay has elapsed the workflow moves onto the next linked stage(s).

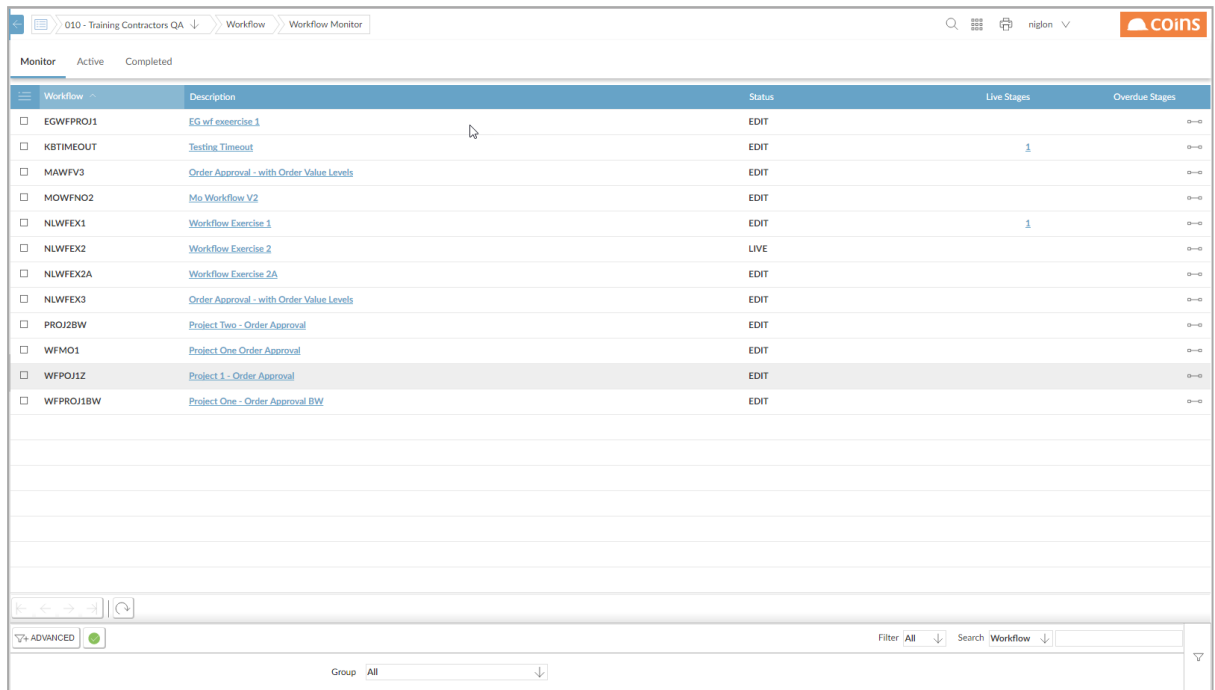
### 8.11.1 Workflow Monitor Workbench

The Workflow Monitor Workbench allows System Administrators and other key users to review and monitor workflow progression

There are three tabs

- Monitor - A Summary of Live and Overdue Stages on those Workflows where the Monitor Status has been checked.
- Active- A List of all workflows that are currently active and awaiting an action to complete.
- Completed - A List of all completed workflows.


### 8.11.1.1 WF Monitor - Monitor Tab



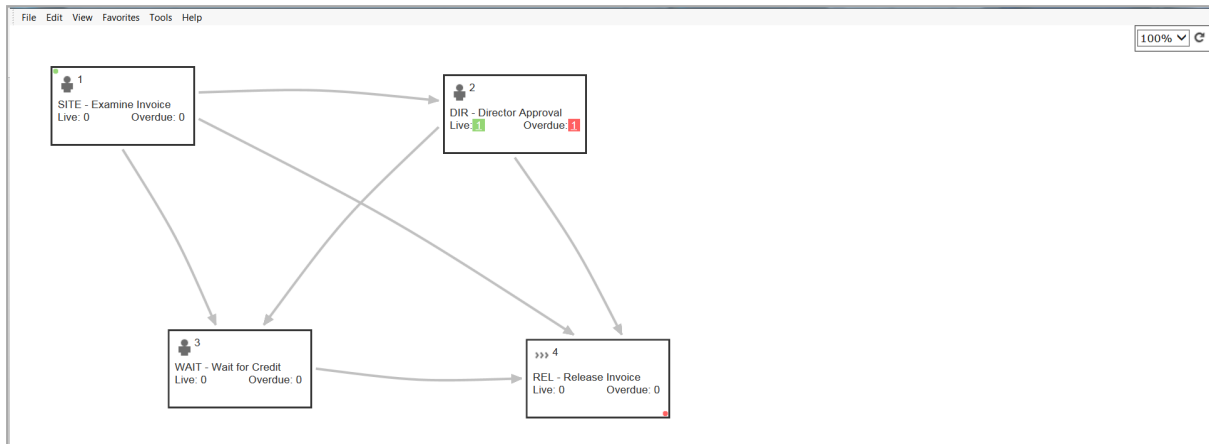
Workflow	Description	Status	Live Stages	Overdue Stages
EGWFFPROJ1	EG.vcf exercise 1	EDIT		
KBTIMEOUT	Testing Timeout	EDIT	1	
MAWFV3	Order Approval - with Order Value Levels	EDIT		
MOWFNO2	Mo Workflow V2	EDIT		
NLWFEX1	Workflow Exercise 1	EDIT	1	
NLWFEX2	Workflow Exercise 2	LIVE		
NLWFEX2A	Workflow Exercise 2A	EDIT		
NLWFEX3	Order Approval - with Order Value Levels	EDIT		
PROJ2BW	Project Two - Order Approval	EDIT		
WFM01	Project One Order Approval	EDIT		
WFPOJ1Z	Project 1 - Order Approval	EDIT		
WFFPROJ1BW	Project One - Order Approval BW	EDIT		

Field	Description
Workflow	Workflow Id
Description	Name of the Workflow
Status	Indicator as to whether the Workflow is LIVE or in EDIT mode
Live Stages	The Number of Stages which are awaiting action on the Activity Workbench
Overdue Stages	The Number of Stages which have exceeded their monitored time limit

#### Workflow Monitor diagram

The workflow monitor has a button  shown on each row to allow the status of the workflow to be shown diagrammatically.

If you select the button then a new window/tab is launched with the diagram shown for the chosen workflow.



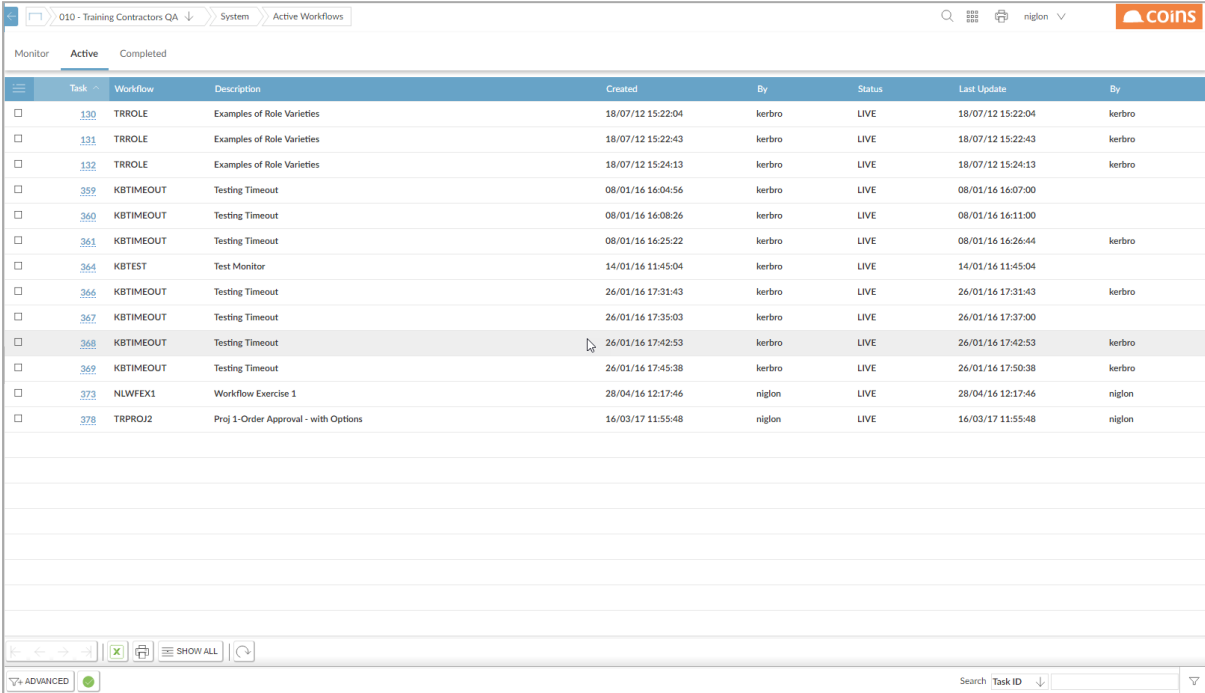
The zoom and refresh options (from the maintenance screen) are available and the diagram shows live and overdue tasks for each of the stages. They are colour coded so that live tasks are shown green and overdue tasks shown red. No tasks are shown as zero and not coloured.

You can drill down to the live and overdue stages by following the link on the numbers shown in the stage.

This is shown in a new window/tab. You can drill further down in the task using the existing task enquiry.

The numbers on the workflow monitor diagram are automatically refreshed every 10 seconds. A change to a task on the activity workbench by users is automatically presented on the monitor diagram.

### 8.11.1.2 WF Monitor - Active Tab

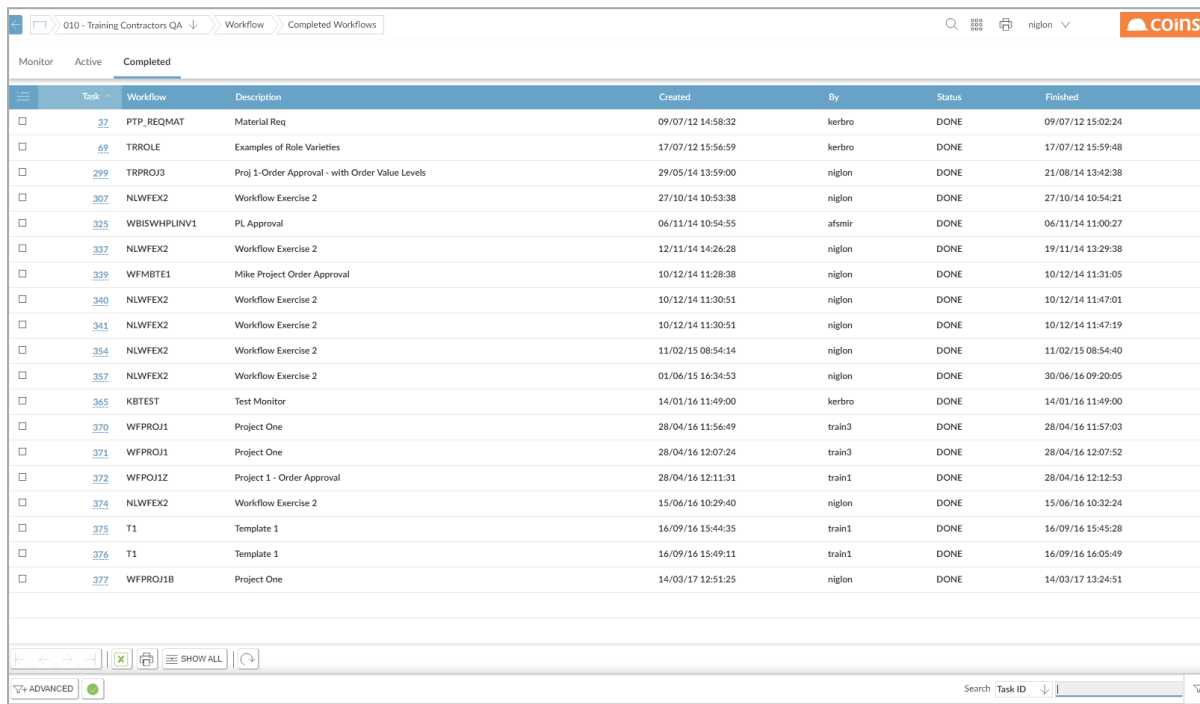


Task	Workflow	Description	Created	By	Status	Last Update	By
130	TRROLE	Examples of Role Varieties	18/07/12 15:22:04	kerbro	LIVE	18/07/12 15:22:04	kerbro
131	TRROLE	Examples of Role Varieties	18/07/12 15:22:43	kerbro	LIVE	18/07/12 15:22:43	kerbro
132	TRROLE	Examples of Role Varieties	18/07/12 15:24:13	kerbro	LIVE	18/07/12 15:24:13	kerbro
359	KBTIMEOUT	Testing Timeout	08/01/16 16:04:56	kerbro	LIVE	08/01/16 16:07:00	
360	KBTIMEOUT	Testing Timeout	08/01/16 16:08:26	kerbro	LIVE	08/01/16 16:11:00	
361	KBTIMEOUT	Testing Timeout	08/01/16 16:25:22	kerbro	LIVE	08/01/16 16:26:44	kerbro
364	KBTEST	Test Monitor	14/01/16 11:45:04	kerbro	LIVE	14/01/16 11:45:04	
366	KBTIMEOUT	Testing Timeout	26/01/16 17:31:43	kerbro	LIVE	26/01/16 17:31:43	kerbro
367	KBTIMEOUT	Testing Timeout	26/01/16 17:35:03	kerbro	LIVE	26/01/16 17:37:00	
368	KBTIMEOUT	Testing Timeout	26/01/16 17:42:53	kerbro	LIVE	26/01/16 17:42:53	kerbro
369	KBTIMEOUT	Testing Timeout	26/01/16 17:45:38	kerbro	LIVE	26/01/16 17:50:38	kerbro
373	NLWFEX1	Workflow Exercise 1	28/04/16 12:17:46	niglon	LIVE	28/04/16 12:17:46	niglon
378	TRPROJ2	Proj 1-Order Approval - with Options	16/03/17 11:55:48	niglon	LIVE	16/03/17 11:55:48	niglon

Field	Description
Task	The internal task number of the workflow instance. This field links to the Workflow Summary
Workflow	Workflow template code
Description	The workflow template description
Created	The date the instance was created
By	Who created the workflow instance
Status	The status of the workflow instance
Last update	The date/time of the workflow instance
By	The COINS userID of the user.

### 8.11.1.3 WF Monitor - Completed Tab

#### Completed Tab

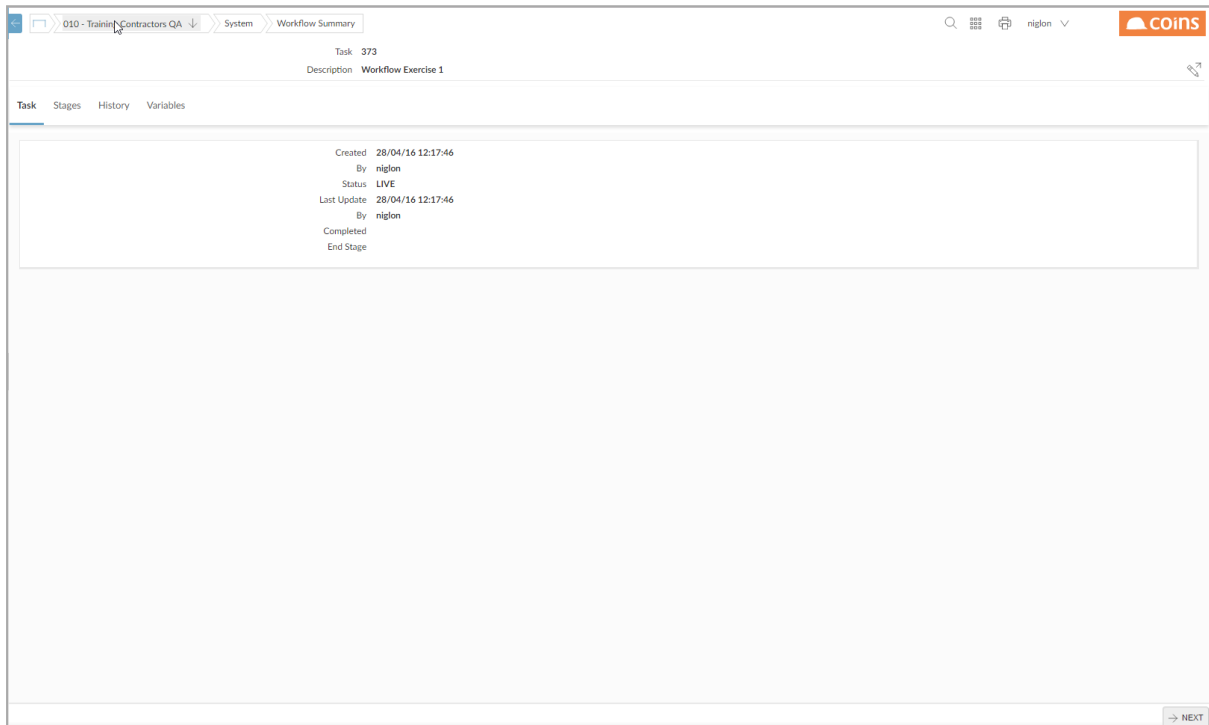


Task	Workflow	Description	Created	By	Status	Finished
37	PTP_REQMAT	Material Req	09/07/12 14:58:32	kerbro	DONE	09/07/12 15:02:24
69	TRROLE	Examples of Role Varieties	17/07/12 15:56:59	kerbro	DONE	17/07/12 15:59:48
299	TRPROJ3	Proj 1-Order Approval - with Order Value Levels	29/05/14 13:59:00	niglon	DONE	21/08/14 13:42:38
307	NLWFEX2	Workflow Exercise 2	27/10/14 10:53:38	niglon	DONE	27/10/14 10:54:21
325	WBISWHPLINV1	PL Approval	06/11/14 10:54:55	afsmir	DONE	06/11/14 11:00:27
337	NLWFEX2	Workflow Exercise 2	12/11/14 14:26:28	niglon	DONE	19/11/14 13:29:38
339	WFMTE1	Mike Project Order Approval	10/12/14 11:28:38	niglon	DONE	10/12/14 11:31:05
340	NLWFEX2	Workflow Exercise 2	10/12/14 11:30:51	niglon	DONE	10/12/14 11:47:01
341	NLWFEX2	Workflow Exercise 2	10/12/14 11:30:51	niglon	DONE	10/12/14 11:47:19
354	NLWFEX2	Workflow Exercise 2	11/02/15 08:54:14	niglon	DONE	11/02/15 08:54:40
357	NLWFEX2	Workflow Exercise 2	01/06/15 16:34:53	niglon	DONE	30/06/16 09:20:05
365	KBTEST	Test Monitor	14/01/16 11:49:00	kerbro	DONE	14/01/16 11:49:00
370	WFPROJ1	Project One	28/04/16 11:56:49	train3	DONE	28/04/16 11:57:03
371	WFPROJ1	Project One	28/04/16 12:07:24	train3	DONE	28/04/16 12:07:52
372	WFPROJ1Z	Project 1 - Order Approval	28/04/16 12:11:31	train1	DONE	28/04/16 12:12:53
374	NLWFEX2	Workflow Exercise 2	15/06/16 10:29:40	niglon	DONE	15/06/16 10:32:24
375	T1	Template 1	16/09/16 15:44:35	train1	DONE	16/09/16 15:45:28
376	T1	Template 1	16/09/16 15:49:11	train1	DONE	16/09/16 16:05:49
377	WFPROJ1B	Project One	14/03/17 12:51:25	niglon	DONE	14/03/17 13:24:51

Field	Description
Task	The internal task number of the workflow instance. This field links to the Workflow Summary
Workflow	Workflow template code
Description	The workflow template description
Created	The date the instance was created
By	Who created the workflow instance
Status	The status of the workflow instance
Finished	The date/time the workflow instance completed

## 8.12 Workflow Summary

From the Active or Completed tab, clicking on the Task number will link to the Workflow summary screen.



Four tabs are available:

- Task
- Stages
- History
- Variables



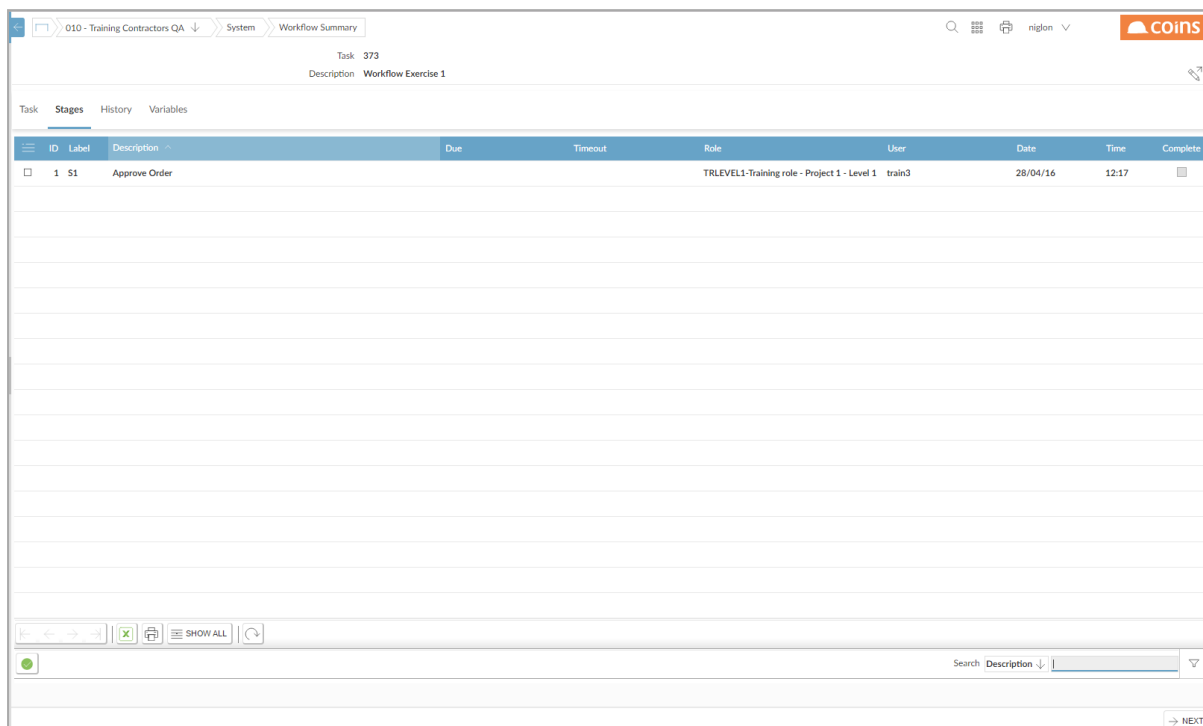


### 8.12.1 WF Summary - Tasks Tab

Task: 114 Description: Proj 1-Order Approval - with Options HM-ng1/0089			
Task	Stages	History	Variables
Created: 18/07/12 12:28:09 By: kerbro Status: LIVE Last Update: 18/07/12 12:28:09 By: kerbro Completed: End Stage:			

Field	Description
Task	The internal task number of the workflow instance.
Description	The workflow template description and the user defined task ID as configured on the workflow template.
Created	The date and time the instance was created
By	Who created the workflow instance
Status	The status of the workflow instance
Last Update	The date/time that the workflow instance was last updated.
By	The CONS user ID of the user
Completed	The date/time the workflow instance was completed
End Stage	The label of the stage that completed the workflow

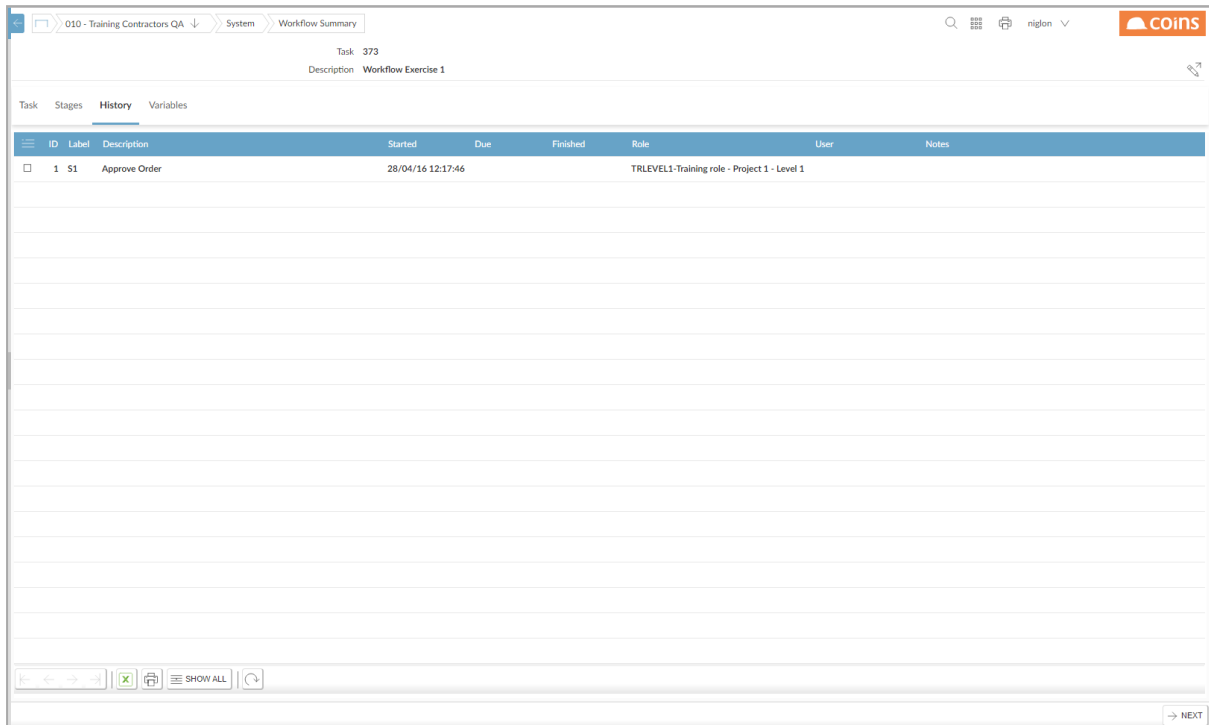
## 8.12.2 WF Summary - Stages Tab



Field	Description
Description	The description of the stage
Due	The date/time the stage is due to be completed
Timeout	The date/time this stage will timeout
User	The user that this appointment or task is for.
Date	The date for the appointment or task.
Time	Time that the action was created/due.
Complete	Whether the action has been completed



### 8.12.3 WF Summary - History Tab



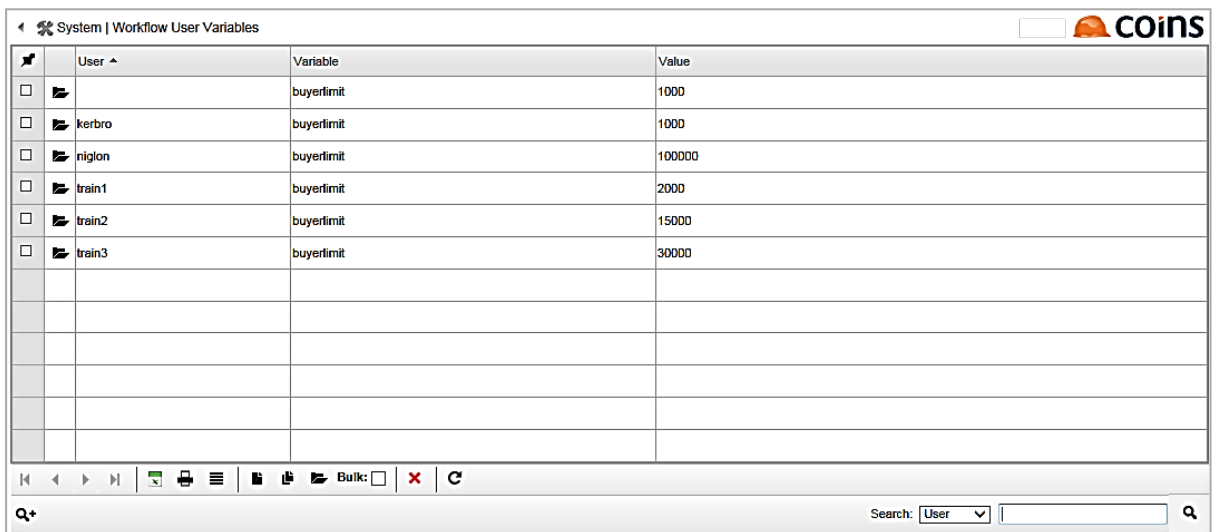
Field	Description
Description	The description of the stage
Started	The date/time the stage started
Due	The date/time that the stage was due to be completed (if the stage had a duration associated with it).
Finished	Date/Time the stage finished
User	The user who completed the stage.
Notes	Any notes entered by the user on the task.



## 8.13 Workflow User Variables

Workflow User Variables allow you to use different values for different users in conditions and calculations. For example, a condition may rely on an authorisation limit which is different for different users.

It is possible to define either general or user based variables for use within Workflow.



	User ^	Variable	Value
<input type="checkbox"/>		buyerlimit	1000
<input type="checkbox"/>	kerbro	buyerlimit	1000
<input type="checkbox"/>	niglon	buyerlimit	100000
<input type="checkbox"/>	train1	buyerlimit	2000
<input type="checkbox"/>	train2	buyerlimit	15000
<input type="checkbox"/>	train3	buyerlimit	30000

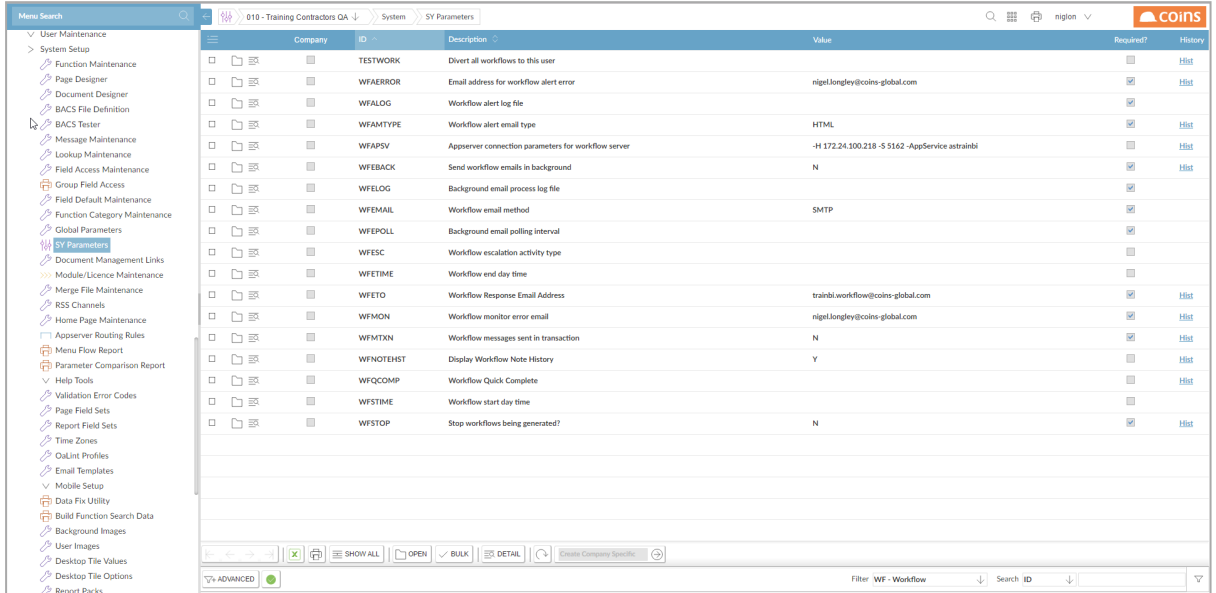
To do this define the User ID (or leave blank for all Users), the variable code and the variable value.

Once a role has been calculated on any stage the User Variable can be called using the following field on the sysuser record:

```
sysuser. RO_sur_wfvar^{VariableCode}
```

## 8.14 Workflow Parameters (System Parameters)

The following system parameters relate to using workflow.



Company	ID	Description	Value	Required?	History
	TESTWORK	Divert all workflows to this user		<input type="checkbox"/>	Hist
	WFAEROR	Email address for workflow alert error	nigel.jongley@coins-global.com	<input checked="" type="checkbox"/>	Hist
	WFALOG	Workflow alert log file		<input checked="" type="checkbox"/>	Hist
	WFAMTYPE	Workflow alert email type	HTML	<input checked="" type="checkbox"/>	Hist
	WFAPSV	Appserver connection parameters for workflow server	-H 172.24.100.218 -S 5162 -AppService astrainbl	<input type="checkbox"/>	Hist
	WFEBACK	Send workflow emails in background	N	<input checked="" type="checkbox"/>	Hist
	WFEOG	Background email process log file		<input checked="" type="checkbox"/>	Hist
	WFEMAIL	Workflow email method	SMTP	<input checked="" type="checkbox"/>	Hist
	WFEPOLL	Background email polling interval		<input checked="" type="checkbox"/>	Hist
	WFESC	Workflow escalation activity type		<input type="checkbox"/>	Hist
	WFETIME	Workflow end day time		<input type="checkbox"/>	Hist
	WFETO	Workflow Response Email Address	trainbl.workflow@coins-global.com	<input checked="" type="checkbox"/>	Hist
	WFEMON	Workflow monitor error email	nigel.jongley@coins-global.com	<input checked="" type="checkbox"/>	Hist
	WFMTXN	Workflow messages sent in transaction	N	<input checked="" type="checkbox"/>	Hist
	WFNOTEHST	Display Workflow Note History	Y	<input type="checkbox"/>	Hist
	WFQCOMP	Workflow Quick Complete		<input checked="" type="checkbox"/>	Hist
	WFSTIME	Workflow start day time		<input type="checkbox"/>	Hist
	WFSTOP	Stop workflows being generated?	N	<input checked="" type="checkbox"/>	Hist





	Table ^	Before	After
<input type="checkbox"/>	pi_project	\$New = inlist(pi_project.pij_num,"");	ifMethod(New,'wfuwfp01.startWorkflow','MK_NewOppS');

In the example above, the table pi\_project is tested for a value in pij\_num for a blank. The implication of this test is that pij\_num is only blank when a record is created. No test is made after the record is changed as the WF is intended to run for each New record.

	Table ^	Before	After
<input type="checkbox"/>	pp_organisation	debug(1);\$old=pp_organisation.ppo_leave-date;	\$new=pp_organisation.ppo_leave-date; b=if(equal(new,old),eq,1,0,1); c=inlist(old,""); ifEXEC(b * c,EQ,1) method (*wfuwfp01.startWorkflow','HRPROJ1');

In the second example above, pp\_organisation is checked for a value in ppo\_leave\_date and the value stored in variable \$old. After the record is changed, the current value of ppo\_leave\_date is stored in \$new and old and new variables compared. If they are the same, nothing happens, but if the leaving date has changed, the workflow is triggered.

Two examples of Database Triggers:

This is a nice simple one to test the triggers are firing and the App Server is running - if you take the test User record and set the name to 'TEST' it will error when you save it.

Table	Before	After
sysuser		ifExec(equal (sysuser.su-name,'Test'),EQ,1 error (1,'Name cannot be Test');

This is a simple Project Trigger to send an alert via email when the field pij\_our\_value is updated.

Table	Before	After
pi_project	debug(1);a=pi_project.pij_our_value;	b=pi_project.pij_our_value;c=if(a,ne,b); ifMethod (c,'sysualert.sendEmail','user@companyl.com','','My Project Value has Changed','Project Number - ' + pi_project.pij_dispno + ' ' + pi_project.pij_name_;

It also has debug switched on so you will see the calculation in the App Server Log File (in a similar way to the Report log file). The Log file can be found in \$BASE/var/diag and is usually name something like:





`oaliveas.server.log`

This can be a huge log file so I suggest testing and debugging on a non-live environment where there are fewer users - preferably just yourself.

## 8.16 14 Import/Export Workflow Template

All elements of the Workflow configuration can be updated using either standard Bulk maintenance options or the import and export options.

## 8.17 15 Launching Workflow

There is a standard method available for launching workflows from within the OA framework using the standard calculation syntax, this is as follows :

```
ifexec(var,EQ,1) method('wfuwfp01.startWorkFlow','WORKFLOWCODE');
```

There is a standard method available for a bulk change of status to a group of records from within the OA framework using the standard calculation syntax, this is as follows:

```
Method('wfuwfp01.completeWorkflowForRowid','{xx_xxxRowid}','  
WORKFLOWCODE', 'Status From','Status To','Notes for history')
```

```
Method('wfuwfp01.completeWorkflowForRowid','{ap_invoiceRowid}','TIM',  
'*', 'A','Notes for history')
```

In the above example this method will update - All invoices within query for workflow named 'TIM' will convert 'all status' to 'status A' and adding a Note to the History. The activity WB task will be completed if it belongs to the person doing the completion otherwise it will be removed.

This can be used in reports, on pages or from a database trigger, indeed anywhere where a calculation is performed.



## 8.19 17 Method to force the closure of Workflow:

There is a method which can be used within the context of a report or within the Calculation editor if it is applicable for one record. The syntax for the method is as follows:

```
method('wfuwfp01.completeWorkFlowForRowid','Rowid','WFName','FromStage','To  
Stage','Comment');
```

Example of use:

```
method('wfuwfp01.completeWorkFlowForRowid','0x0000000000443c85','EX_  
1','*', 'END','Forcing the Workflow to close via method calc');
```

## 9 Introduction or Executive Summary

This document describes the capability for the user to define services using existing components of COINS.

The user is able to define the following general capabilities using OA designer:

- Datasets – to extract data records for use on reports
- Page Maintenance – which can include add, update and delete capabilities
- Calculation Programs – that can run low level methods in COINS RSPs to extract and/or manipulate data

It is now possible for the user to also define an interface that matches up to these generic features so that they can be run as a service to be exposed on an Enterprise Service Bus (ESB) or as a Webservice.

## 9.1 soapUI

The examples given in this document use the Open Source tool soapUI to demonstrate the interaction of the Web Services. It is assumed that clients working with Web Services will have their own tools and applications, but if you wish to follow the examples directly, soapUI can be downloaded from.:

<http://www.soapui.org>

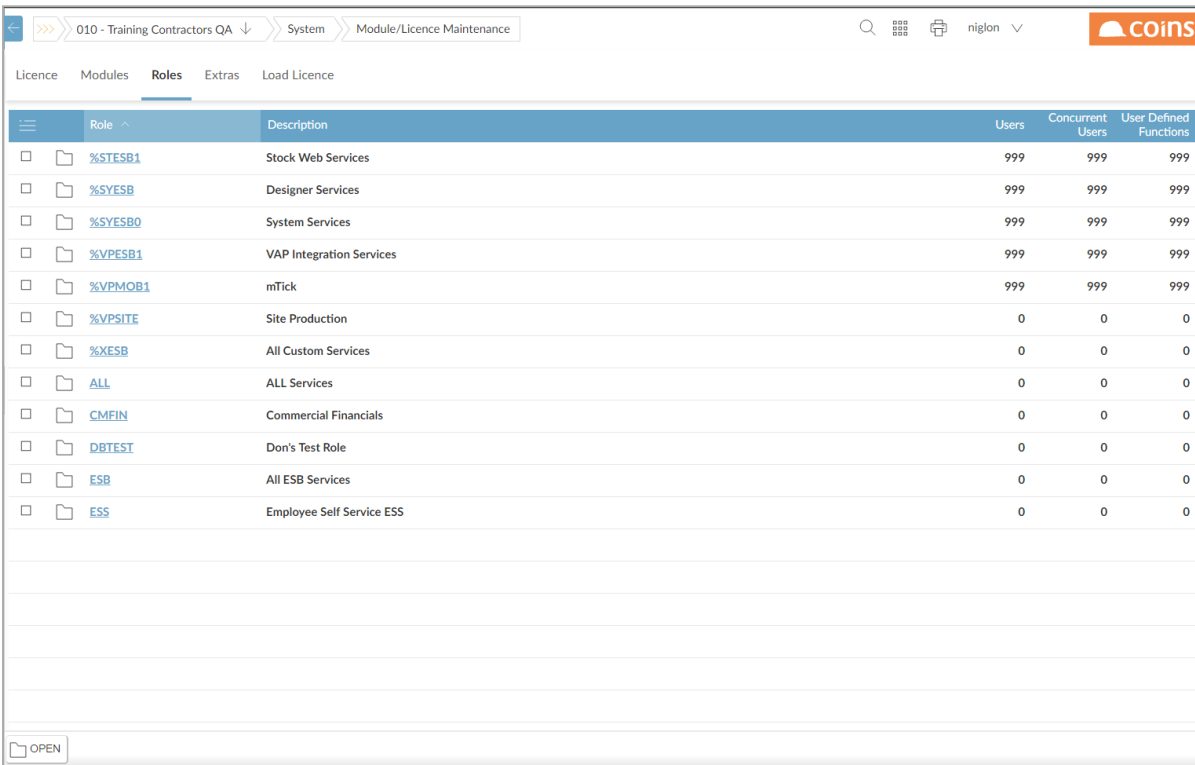
## 9.2 Licencing

If you encounter any messages regarding licencing whilst using Web Services, check the following:

### 9.2.1 Additional Licencing

User Defined Services require an additional element of licencing.

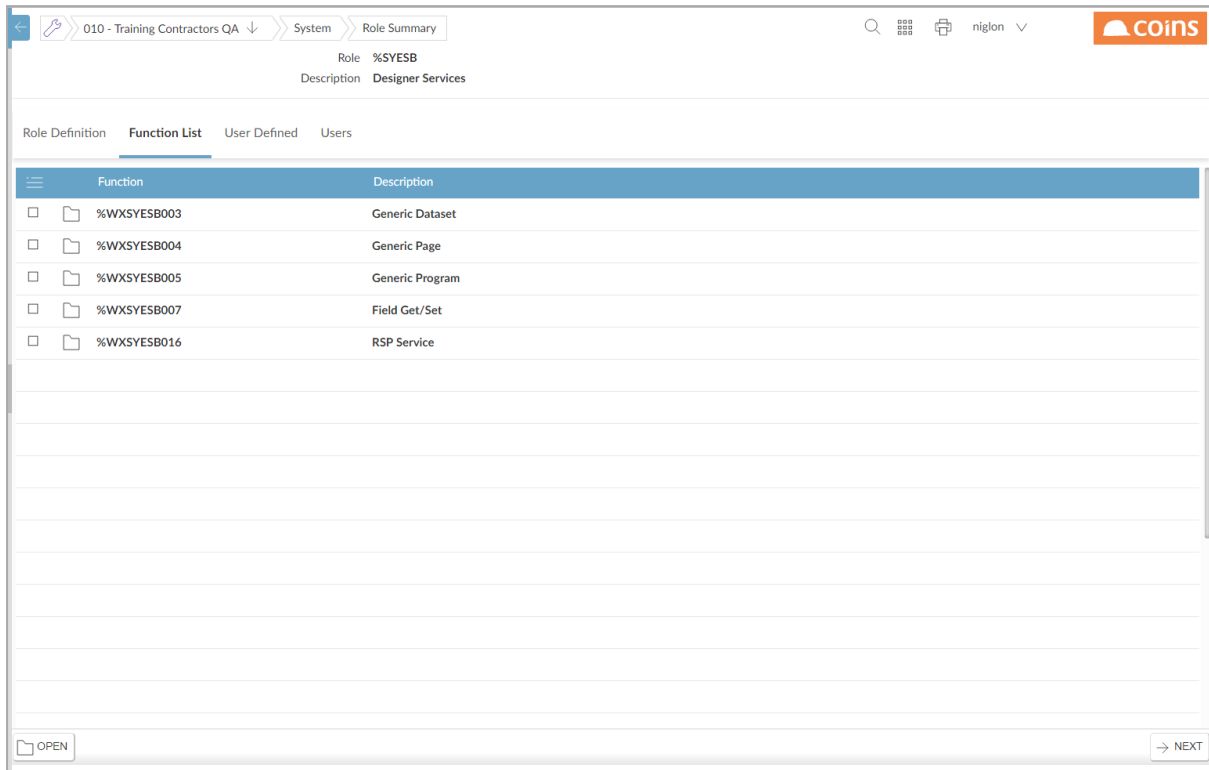
In Module/Licence Maintenance, the Role %SYESB must be present



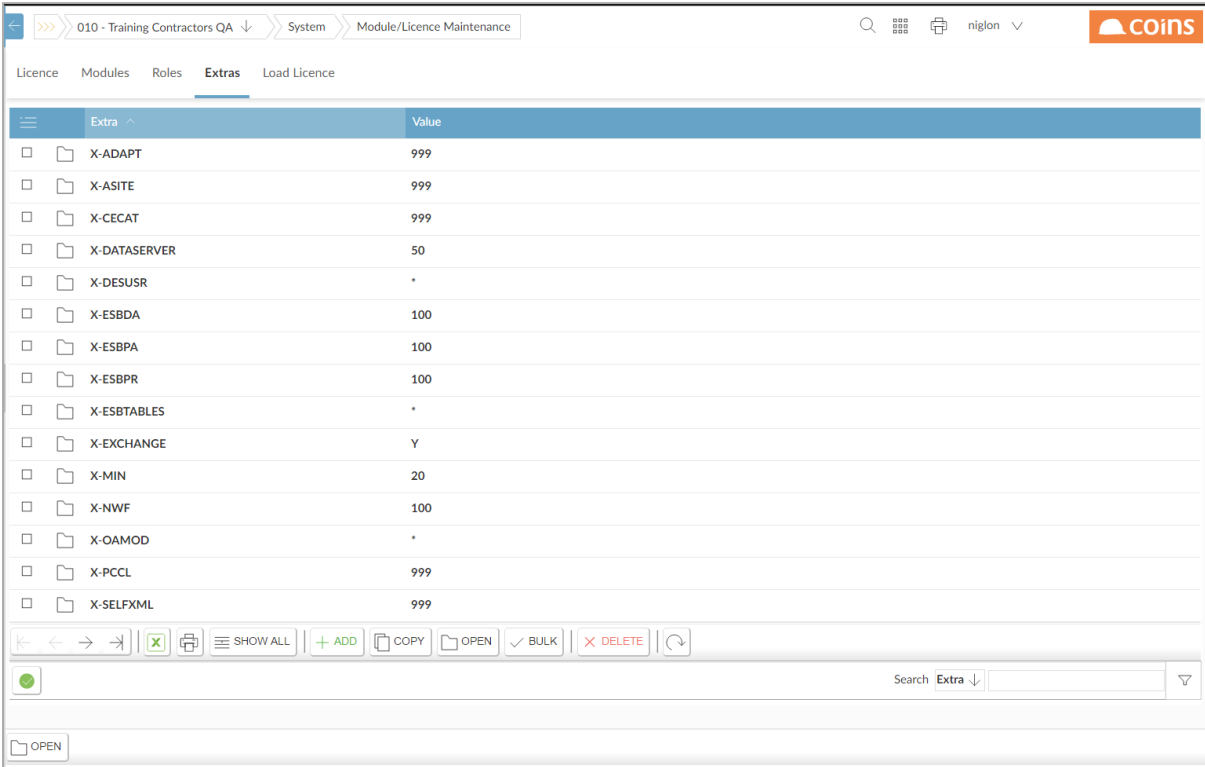
Role	Description	Users	Concurrent Users	User Defined Functions
<input type="checkbox"/> %STESB1	Stock Web Services	999	999	999
<input type="checkbox"/> %SYESB	Designer Services	999	999	999
<input type="checkbox"/> %SYESB0	System Services	999	999	999
<input type="checkbox"/> %VPESB1	VAP Integration Services	999	999	999
<input type="checkbox"/> %VPMOB1	mTick	999	999	999
<input type="checkbox"/> %VPSITE	Site Production	0	0	0
<input type="checkbox"/> %XESB	All Custom Services	0	0	0
<input type="checkbox"/> ALL	ALL Services	0	0	0
<input type="checkbox"/> CMFIN	Commercial Financials	0	0	0
<input type="checkbox"/> DBTEST	Don's Test Role	0	0	0
<input type="checkbox"/> ESB	All ESB Services	0	0	0
<input type="checkbox"/> ESS	Employee Self Service ESS	0	0	0

Within this, the functions %WXSYESB003/004/005 and 007 should be assigned:





Once you have checked that these are in place. Under the module/Licence Maintenance Extras Tab, there are four entries that define which user-defined services can be used:

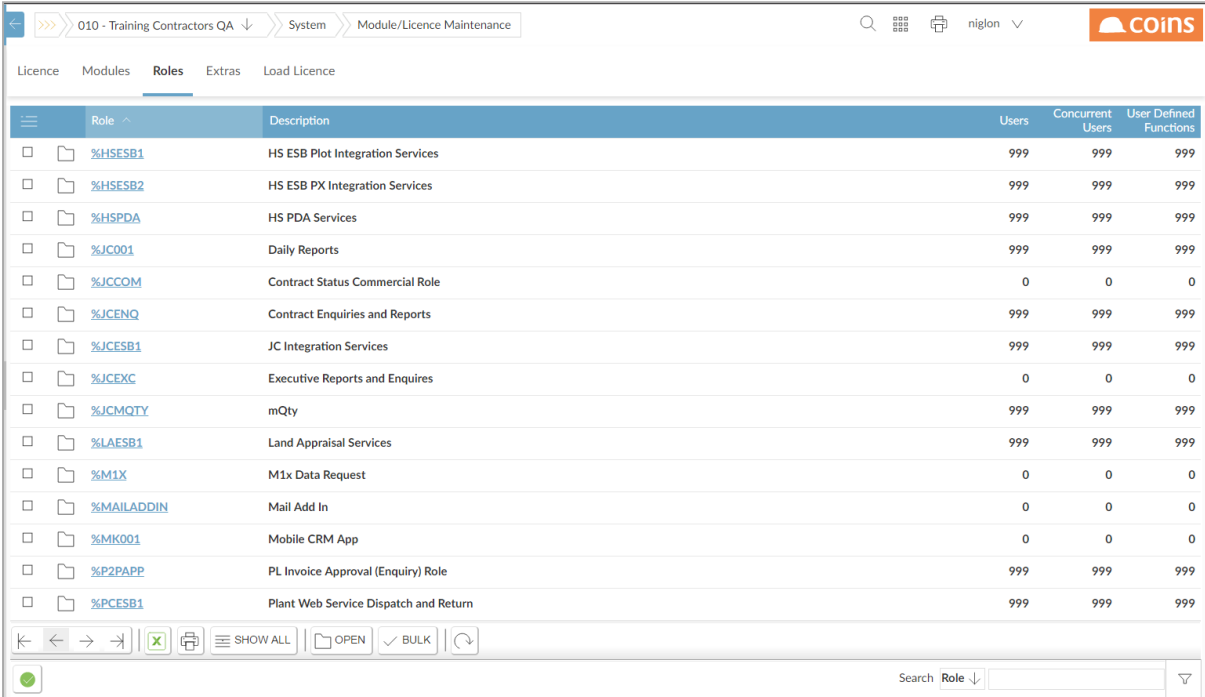


Extra	Value
X-ADAPT	999
X-ASITE	999
X-CECAT	999
X-DATASERVER	50
X-DESUSR	*
X-ESBDA	100
X-ESBPA	100
X-ESBPR	100
X-ESBTABLES	*
X-EXCHANGE	Y
X-MIN	20
X-NWF	100
X-OAMOD	*
X-PCCL	999
X-SELFXML	999

- X-ESBDA controls how many datasets can be created within user defined services.
- X-ESBPA controls the number of pages that can be built
- X-ESBPR controls how many user defined programs
- X-ESBTABLES controls how many GET-SET services may be defined.

## 9.2.2 Roles

Web Services are licenced using Roles. These may be found within System Setup/Module Licence Maintenance. The Users column will indicate the number of users who may call the service – typically these will be named users.



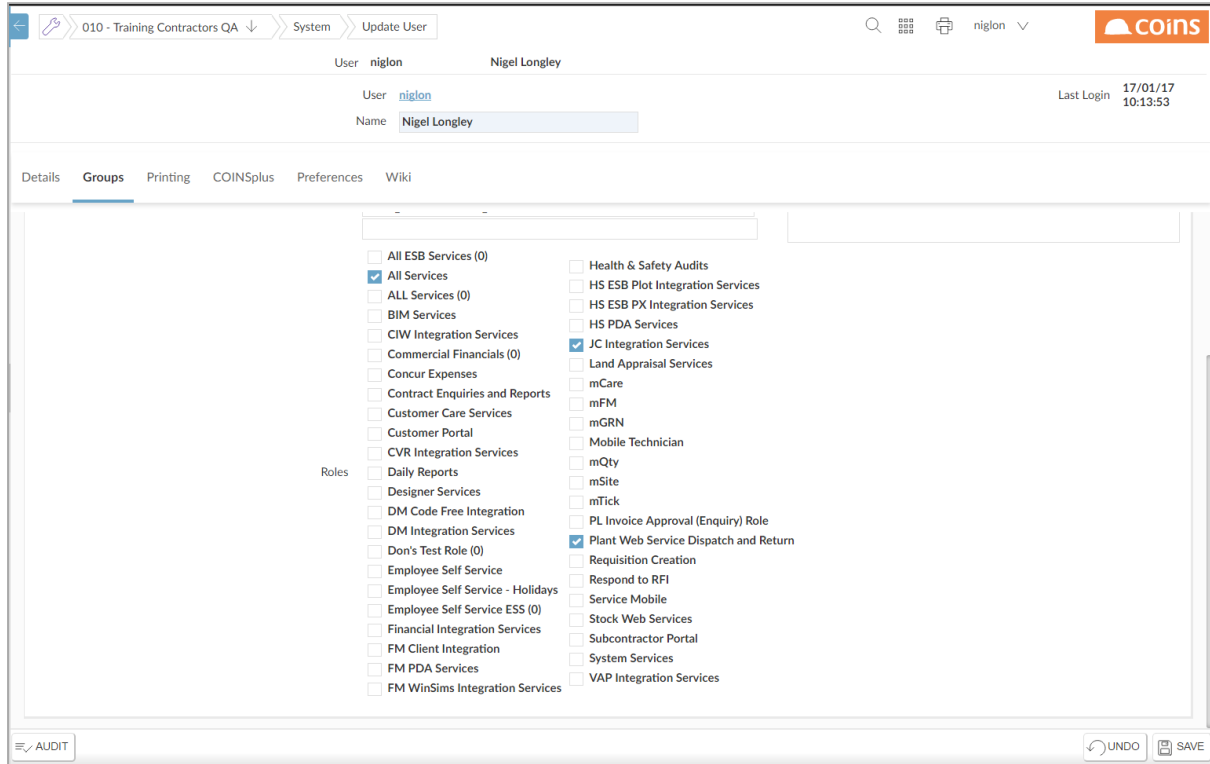
Role	Description	Users	Concurrent Users	User Defined Functions
%HSESB1	HS ESB Plot Integration Services	999	999	999
%HSESB2	HS ESB PX Integration Services	999	999	999
%HSPDA	HS PDA Services	999	999	999
%JC001	Daily Reports	999	999	999
%JCCOM	Contract Status Commercial Role	0	0	0
%JCENQ	Contract Enquiries and Reports	999	999	999
%JCESB1	JC Integration Services	999	999	999
%JCEXC	Executive Reports and Enquires	0	0	0
%JCMQTY	mQty	999	999	999
%LAESB1	Land Appraisal Services	999	999	999
%M1X	M1x Data Request	0	0	0
%MAILADDIN	Mail Add In	0	0	0
%MK001	Mobile CRM App	0	0	0
%P2PAPP	PL Invoice Approval (Enquiry) Role	999	999	999
%PCESB1	Plant Web Service Dispatch and Return	999	999	999

Each role needs to be set up to give access to specific service functions



### 9.2.3 User Maintenance

To grant access to the required service, open the User record within System/User Maintenance and select the Groups Tab.



Tick the appropriate Roles – subject to the licensing granted to your company and click Save.

## 9.3 Viewing Installed Services

To view the services installed on your system, take your standard environment access URL, such as:

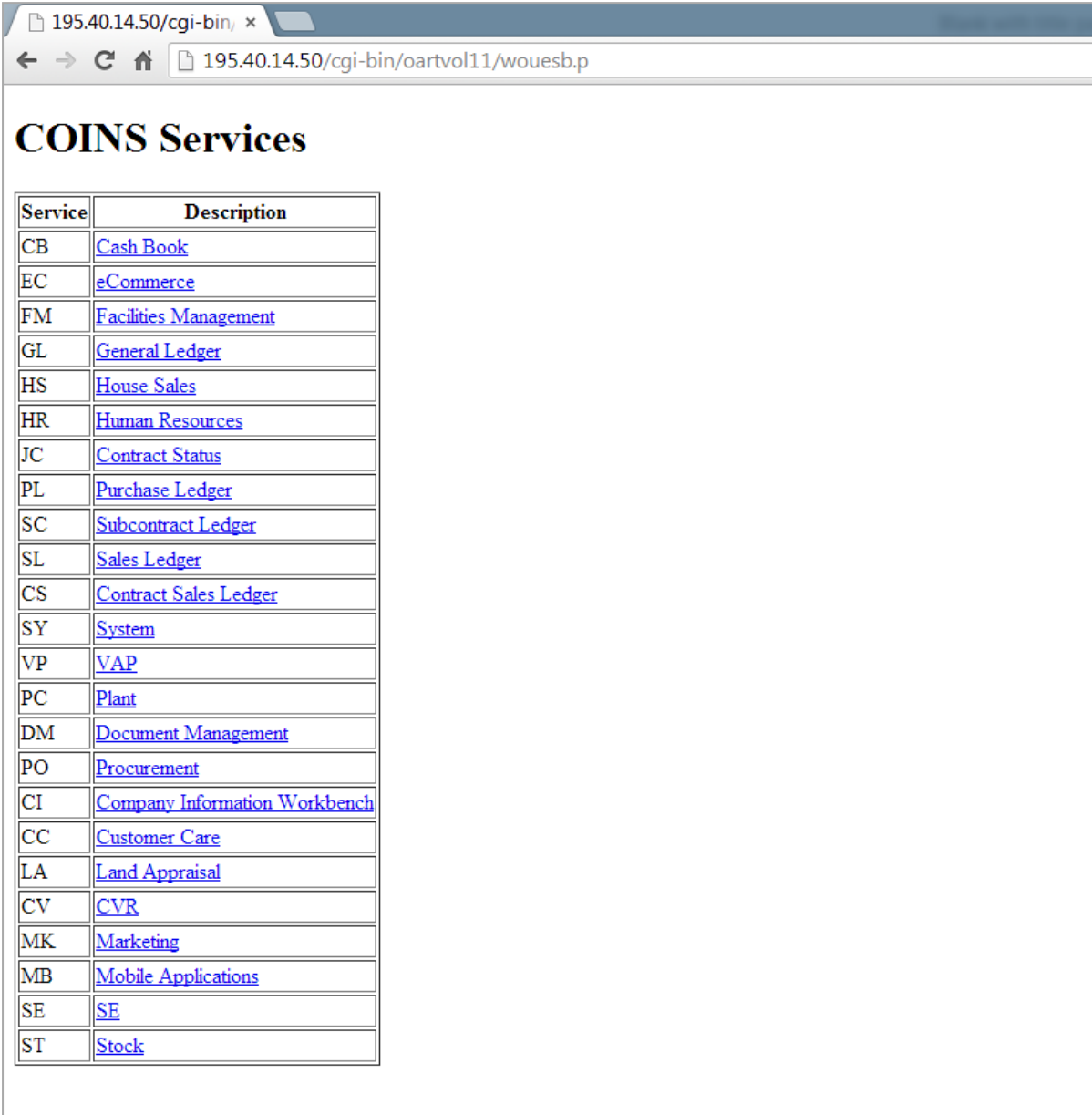
`http://195.40.14.50/cgi-bin/oartvol11/wologin.p`

and replace `wologin.p` (and anything after it) with `wouesp.p`

For example:

`http://195.40.14.50/cgi-bin/oartvol11/wouesb.p`

The following page should be displayed.



Service	Description
CB	<a href="#">Cash Book</a>
EC	<a href="#">eCommerce</a>
FM	<a href="#">Facilities Management</a>
GL	<a href="#">General Ledger</a>
HS	<a href="#">House Sales</a>
HR	<a href="#">Human Resources</a>
JC	<a href="#">Contract Status</a>
PL	<a href="#">Purchase Ledger</a>
SC	<a href="#">Subcontract Ledger</a>
SL	<a href="#">Sales Ledger</a>
CS	<a href="#">Contract Sales Ledger</a>
SY	<a href="#">System</a>
VP	<a href="#">VAP</a>
PC	<a href="#">Plant</a>
DM	<a href="#">Document Management</a>
PO	<a href="#">Procurement</a>
CI	<a href="#">Company Information Workbench</a>
CC	<a href="#">Customer Care</a>
LA	<a href="#">Land Appraisal</a>
CV	<a href="#">CVR</a>
MK	<a href="#">Marketing</a>
MB	<a href="#">Mobile Applications</a>
SE	<a href="#">SE</a>
ST	<a href="#">Stock</a>

The contents of this page will vary, depending on the services installed on your particular system.

Please note that this page will be displayed regardless of the licencing installed on your system – this means that although you will be able to view the services, you may not be able to run certain ones if you have not been licenced to do so.

Selecting a Module Service, for example JC – Contract Status, will display the services for that module:

---

## COINS Services

[COINS Services](#) >Contract Status

Service	Description
JCESB001	<a href="#">Contract Section Maintenance</a>
JCESB002	<a href="#">Project/Contract/Site Create</a>
JCESB003	<a href="#">Contract Reconciliation</a>
JCESB004	<a href="#">Costs-Revenue-Debtors-Credit</a>
JCESB005	<a href="#">Cost Transaction Load</a>
JCESB006	<a href="#">Revenue Transaction Load</a>
JCESB007	<a href="#">Contract Changes</a>
JCESB008	<a href="#">Company Changes</a>
JCESB009	<a href="#">Contract Cost Transactions</a>
JCESB010	<a href="#">Contract Section Maintenance</a>
JCESB011	<a href="#">WBS Code Changes</a>
JCESB017	<a href="#">Timesheet Master Data</a>
JCESB015	<a href="#">Contracts and WBS Codes List</a>
JCESB016	<a href="#">jc Mobile Webservice</a>
JCESB018	<a href="#">Cost Code List</a>

If you then drill down into a particular service, For example JCESB008 – Company Changes, you will be taken into a page detailing the schema, field definitions and sample messages for the service.





**COINS Services**

COINS Services > Contract Status > Company Changes

Service	ICESB008
Description	Company Changes
Schema	input.xsd output.xsd exception.xsd acknowledgement.xsd SOAPHeader.xsd SOAPPostreq.xsd
WSDL	ICESB008.wsdl
WSDL NS	ICESB008.wsdl

The companies changes service will return the companies that have changed since the specified date for a selected set of companies.

**Input**

Entity	Type	Documentation
COINSInterface		
Header		
id	string	A unique message identifier from the originating system.
confirm	string	Whether a confirmation message is required. Set to true or yes to have a confirmation message returned. If this is set then an id must also be provided.
action	string	An action type that will indicate why the message was created. For example, this might be CREATE, UPDATE, or DELETE for a message as a result of a maintenance of a record or PUBLISH for the publish of some information.
entity	string	The service that should consume the message.
arguments	string	Arguments that should be passed to the service.
ackID	string	An optional acknowledgement ID. If this ID is set then COINS will respond with an acknowledgement message returning this ackID in the message header.
testMsg	boolean	Whether this message is a test message. A test message will process through in exactly the same way that a normal message would except that at the point where it would normally commit the transaction to the database, the message is then backed out.
UserID	string	The user in the originating system that caused the message to be produced.
From	string	The name of the system that produced the message.
HostName	string	This is the name of the host system (on UNIX the HOSTNAME environment variable). This is automatically checked (if present in the message) and if it does not match the host name of the system then an exception message is created. If the HostName tag is missing then HostName checking is omitted.
Environment	string	This is the name of the COINS environment that is the intended target for the message. This is automatically checked based on the name of the instance of the COINS system if it does not match then an exception message is created.
Created	dateTime	Date/Time of the source system when the message was created.
Version	string	
Login		

If this page does not show the fields and sample messages, Web Services has not been configured and you will need to contact COINS Support to arrange for Tech Services to investigate

## 9.4 Testing a Service Connection (Using soapUI)

From your browser session, using the JC – Contract Service as our example, select the Company Changes option. In the first schema Panel, locate the WSDL entry.

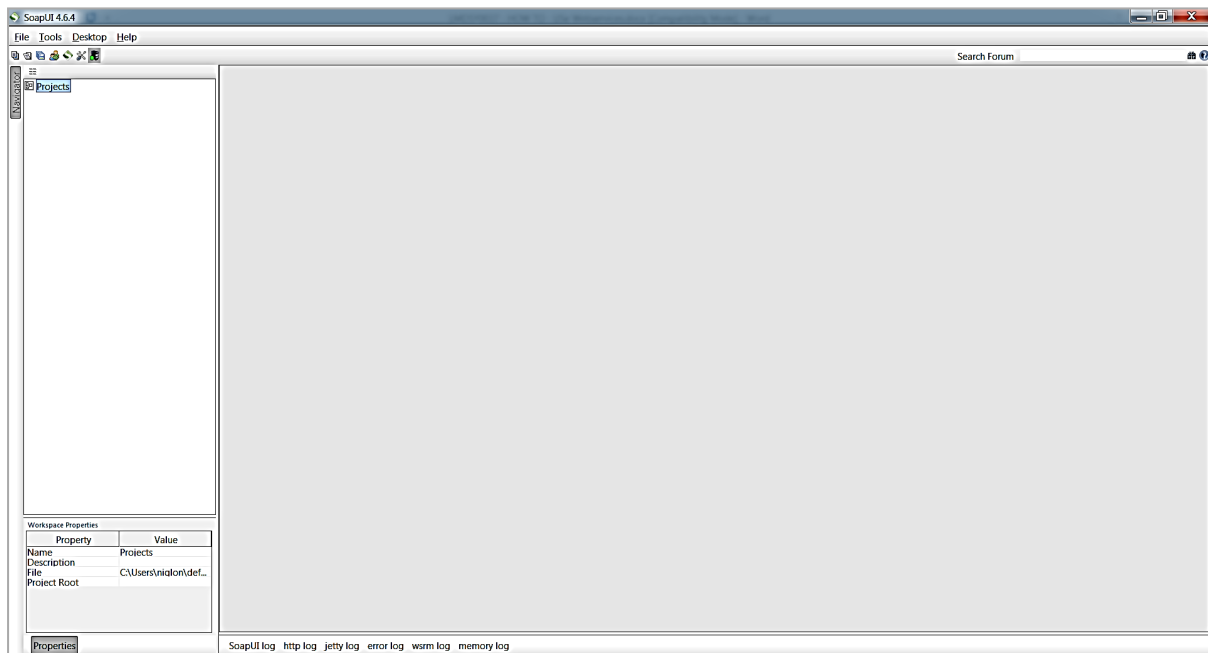
**COINS Services**

[COINS Services](#) > [Job Status](#) > [Company Changes](#)

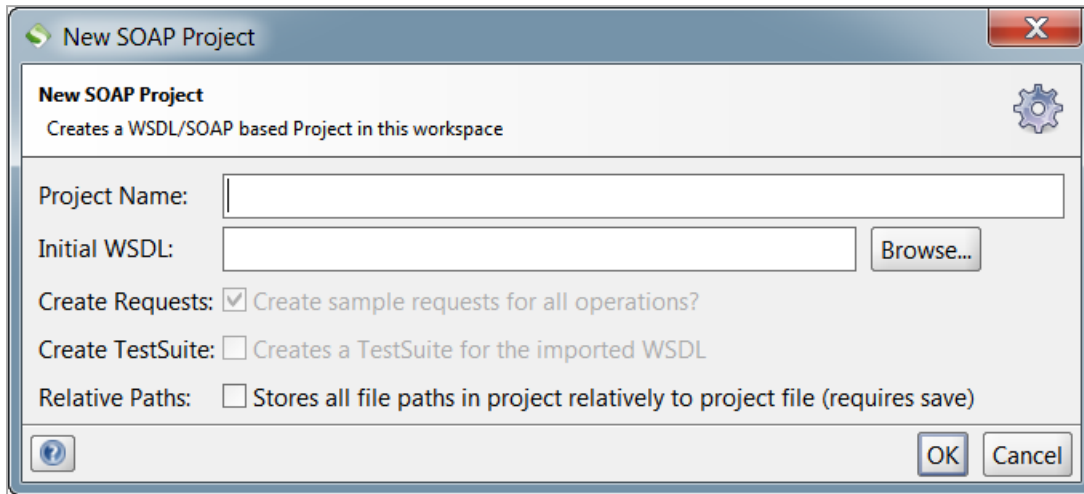
<b>Service</b>	JCESB008
<b>Description</b>	Company Changes
<b>Schema</b>	<a href="#">input.xsd</a> <a href="#">output.xsd</a> <a href="#">exception.xsd</a> <a href="#">acknowledgement.xsd</a> <a href="#">SOAPInput.xsd</a> <a href="#">SOAPOutput.xsd</a>
<b>WSDL</b>	<a href="#">JCESB008.wsdl</a> ←
<b>WSDL NS</b>	<a href="#">JCESB008.wsdl</a>

Right-click on this and select Copy Shortcut.

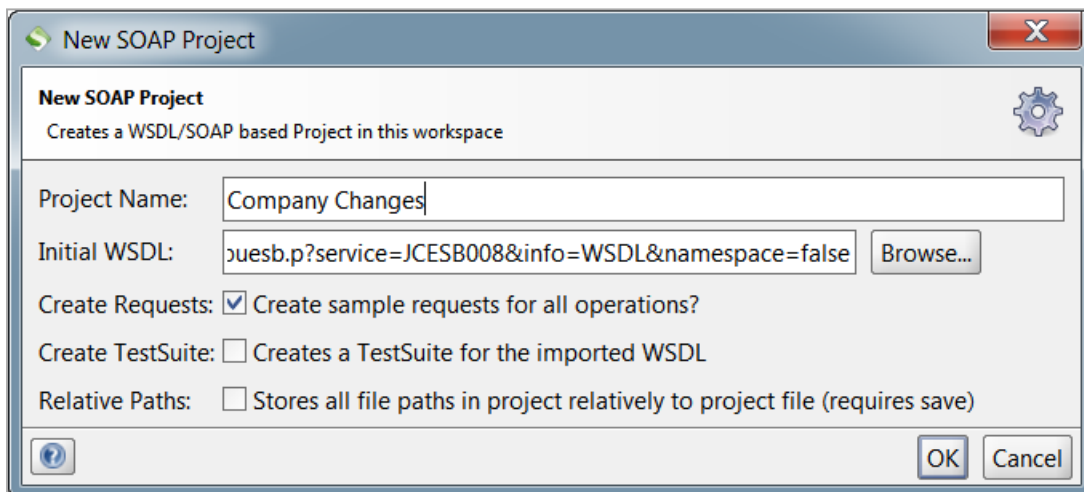
Launch the soapUI application.



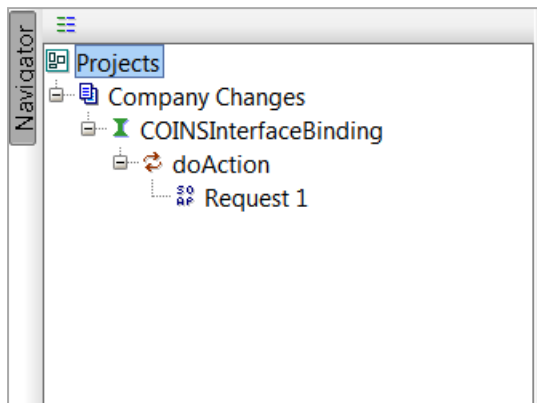
Select File/New SOAP Project.



Paste the shortcut contents into the Initial WSDL: field. Change the Project Name to something more meaningful.



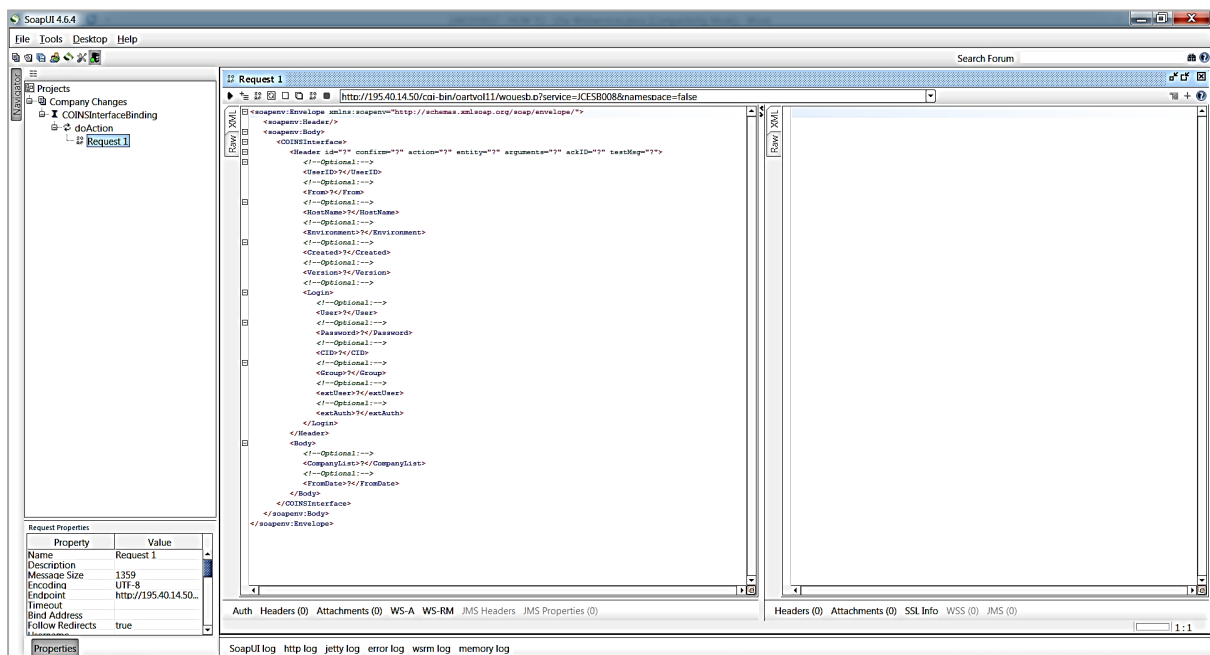
Click OK.



The application will examine the contents of the XML file from the shortcut and will build the necessary interactions.

If successful, the navigator panel should display a result similar to that shown here.

Double click on Request 1 to process the XML and build the appropriate message format.



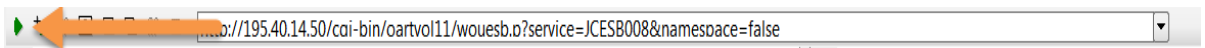
The message details (shown below left) are the same as those generated as Sample Output in the Services Browse (shown below right). The only difference is the version in the soapUI is more verbose with details such as optional fields indicated.

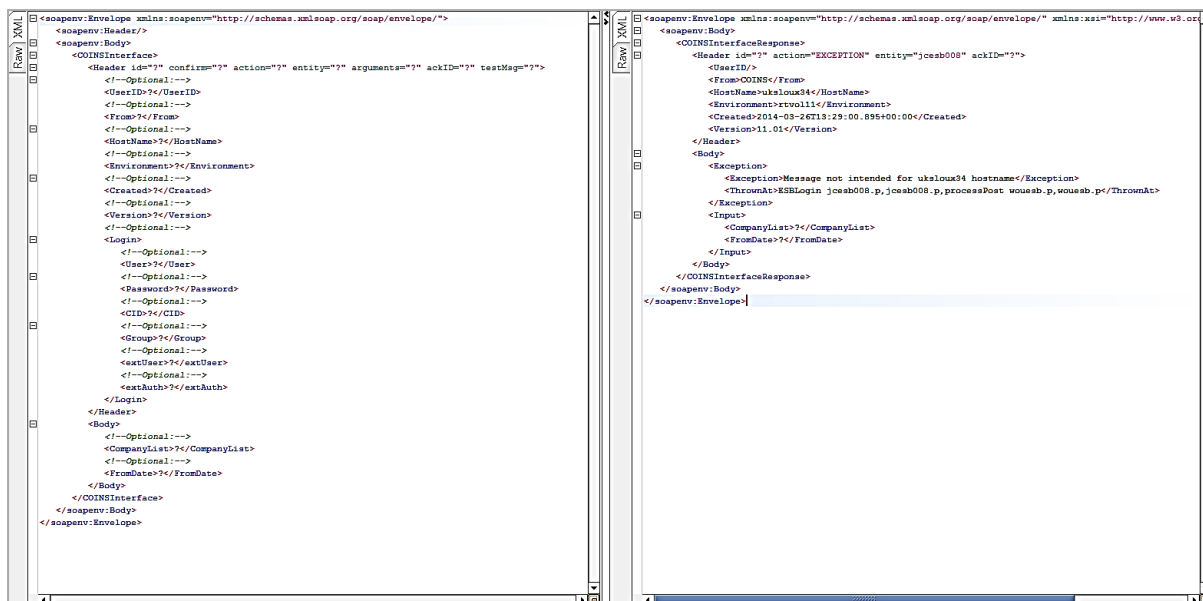
### Sample

```
<COINSInterface>
  <Header id="" confirm="" action="" entity="" arguments="" ackID="" testMsg="true">
    <UserID></UserID>
    <From></From>
    <HostName></HostName>
    <Environment></Environment>
    <Created>2014-03-26T12:58:31.743+00:00</Created>
    <Version></Version>
    <Login>
      <User></User>
      <Password></Password>
      <CID>1</CID>
      <Group></Group>
      <extUser></extUser>
      <extAuth></extAuth>
    </Login>
  </Header>
  <Body>
    <CompanyList></CompanyList>
    <FromDate>2014-03-26</FromDate>
  </Body>
</COINSInterface>
```

```
<?xml version='1.0' encoding='UTF-8'>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <COINSInterface>
      <Header id="?" confirm="?" action="?" entity="?" arguments="?" ackID="?" testMag="?">
        <!--Optional:-->
        <UserID?</UserID>
        <!--Optional:-->
        <From?</From>
        <!--Optional:-->
        <HostName?</HostName>
        <!--Optional:-->
        <Environment?</Environment>
        <!--Optional:-->
        <Created?</Created>
        <!--Optional:-->
        <Version?</Version>
        <!--Optional:-->
        <Login>
          <!--Optional:-->
          <User?</User>
          <!--Optional:-->
          <Password?</Password>
          <!--Optional:-->
          <CID?</CID>
          <!--Optional:-->
          <Group?</Group>
          <!--Optional:-->
          <extUser?</extUser>
          <!--Optional:-->
          <extAuth?</extAuth>
        </Login>
      </Header>
      <Body>
        <!--Optional:-->
        <CompanyList?</CompanyList>
        <!--Optional:-->
        <FromDate?</FromDate>
      </Body>
    </COINSInterface>
  </soapenv:Body>
</soapenv:Envelope>
```

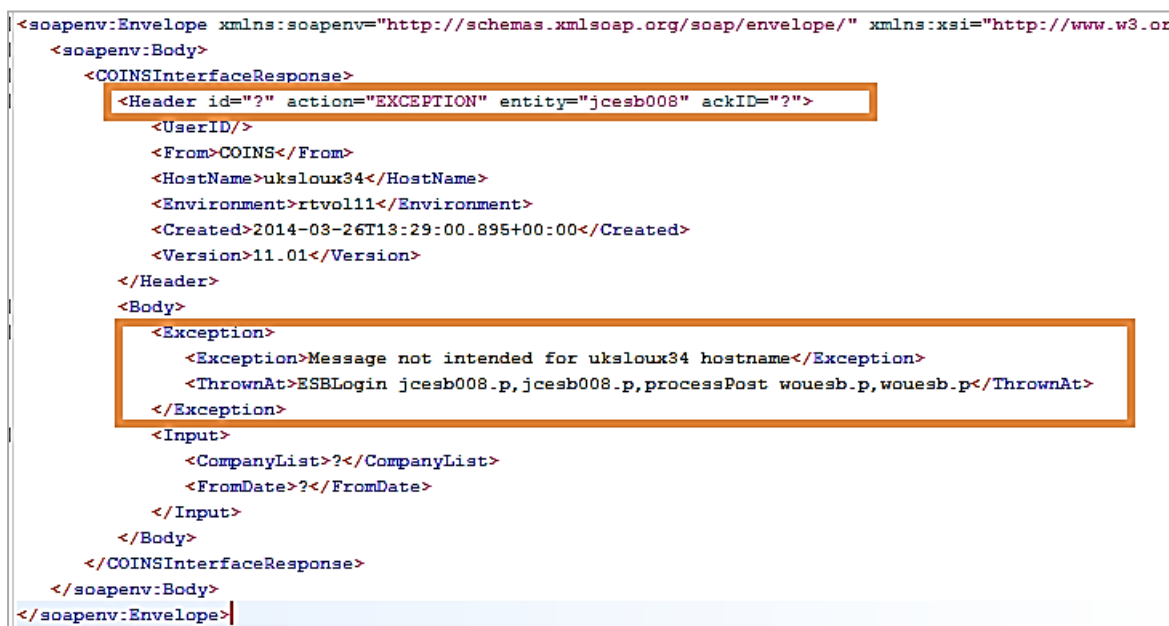
To submit the request message, click the green arrow to the left of the toolbar





The message will be sent to the server and the server will issue an appropriate response – shown in the right hand panel.

Since we sent an empty message in the first instance, the request will fail as shown below:



The exception has been generated because the COINS server expects to be told which server and which environment the message is intended for. A request sent to the wrong server will fail. This is a protection measure to ensure that you do not issue requests to a LIVE environment during testing, or to a TEST environment when you go LIVE.

In our message, therefore, we need to fill in the destination details:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <COINSInterface>
      <Header id="?" confirm="?" action="?" entity="?" arguments="?" ackID="?" testMsg="?">
        <!--Optional:-->
        <UserID?</UserID>
        <!--Optional:-->
        <From?</From>
        <!--Optional:-->
        <HostName>dev.coins-global.com</HostName>
        <!--Optional:-->
        <Environment>dev</Environment>
        <!--Optional:-->
        <Created?</Created>
        <!--Optional:-->
        <Login>
          <!--Optional:-->
          <User?</User>
          <!--Optional:-->
          <Password?</Password>
          <!--Optional:-->
          <CID?</CID>
          <!--Optional:-->
          <Group?</Group>
          <!--Optional:-->
          <extUser?</extUser>
          <!--Optional:-->
          <extAuth?</extAuth>
        </Login>
      </Header>
      <Body>
        <!--Optional:-->
        <CompanyList?</CompanyList>
        <!--Optional:-->
        <FromDate?</FromDate>
      </Body>
    </COINSInterface>
  </soapenv:Body>
</soapenv:Envelope>
```

Once you have made the changes, send the message again.

This time the request should go further, but will fail again due to invalid username/password.





```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <COINSInterfaceResponse>
      <Header id="?" ackID="?" action="EXCEPTION" entity="jcesb008">
        <UserID/>
        <From>COINS</From>
        <HostName>dev.coins-global.com</HostName>
        <Environment>dev</Environment>
        <Created>2014-03-26T13:52:40.566+00:00</Created>
      </Header>
      <Body>
        <Exception>
          <Exception>Invalid User/Password [SY703]</Exception>
          <Inrownat>SSBLogin jcesb008.p,jcesb008.p,processpost wouesb.p,wouesb.p</Inrownat>
        </Exception>
        <Input>
          <CompanyList>?</CompanyList>
          <FromDate>?</FromDate>
        </Input>
      </Body>
    </COINSInterfaceResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Once again, this is a security measure and needs to be specified within the request.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <COINSInterface>
      <Header id="?" confirm="?" action="?" entity="?" arguments="?" ackID="?" testMsg="?">
        <!--Optional:-->
        <UserID?</UserID>
        <!--Optional:-->
        <From?</From>
        <!--Optional:-->
        <HostName>dev.coins-global.com</HostName>
        <!--Optional:-->
        <Environment>dev</Environment>
        <!--Optional:-->
        <Created?</Created>
        <!--Optional:-->
        <Login>
          <!--Optional:-->
          <User>niglon</User>
          <!--Optional:-->
          <Password>abc1123</Password>
          <!--Optional:-->
          <CID?</CID>
          <!--Optional:-->
          <Group?</Group>
          <!--Optional:-->
          <extUser?</extUser>
          <!--Optional:-->
          <extAuth?</extAuth>
        </Login>
      </Header>
      <Body>
        <!--Optional:-->
        <CompanyList?</CompanyList>
        <!--Optional:-->
        <FromDate?</FromDate>
      </Body>
    </COINSInterface>
  </soapenv:Body>
</soapenv:Envelope>
```

Whilst we are making some more changes, we can remove the lines for Group, extUser and extAuth as these are not generally used.

Send the request again, and this time you should get a RESPONSE message instead of an EXCEPTION.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <COINSInterfaceResponse>
      <Header id="?" ackID="?" action="RESPONSE" entity="jcesb008">
        <UserID>NIGLON</UserID>
        <From>COINS</From>
        <HostName>dev.coins-global.com</HostName>
        <Environment>dev</Environment>
        <Created>2014-03-26T14:07:37.837+00:00</Created>
      </Header>
      <Body></Body>
    </COINSInterfaceResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

This indicates that we now have a correct message format.

However, whilst we got a response, the response came back with no data. This is because the service we are using as an example requests details of changes made to COINS company records since a certain date. So we need to add one more change to our request and specify, in the Body entry, which company and from which date. For example:

```
<Body>
  <!--Optional:-->
  <CompanyList>*</CompanyList>
  <!--Optional:-->
  <FromDate>01/01/12</FromDate>
</Body>
</COINSInterface>
</soapenv:Body>
</soapenv:Envelope>
```

Sending the request again now returns another RESPONSE request and some data.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <COINSInterfaceResponse>
      <Header id="?" ackID="?" action="RESPONSE" entity="jcesb008">
        <UserID>NIGLON</UserID>
        <From>COINS</From>
        <HostName>dev.coins-global.com</HostName>
        <Environment>dev</Environment>
        <Created>2014-03-26T14:13:21.464+00:00</Created>
      </Header>
      <Body>
        <co_configRow>
          <kco>1</kco>
          <coc_name>COINS</coc_name>
          <coc_lastrev>2013-05-09</coc_lastrev>
          <coc_vatregno>578524893</coc_vatregno>
          <coc_add1>12 The COINS Budfilding</coc_add1>
          <coc_add2>12 Tdfhe Grove</coc_add2>
          <coc_add3>Slough</coc_add3>
          <coc_add4>Berkshire</coc_add4>
          <coc_pcode>SL1 1QP</coc_pcode>
          <coc_phone>01753 501000</coc_phone>
          <coc_fax>01753 711010</coc_fax>
          <coc_reg>&lt;P>1234567890 324567832&lt;/P></coc_reg>
          <cnt_code>GB</cnt_code>
          <cur_code>GBP</cur_code>
          <coc_contref>607 1440 9</coc_contref>
          <coc_coinsid>2004-08-19T12:15:390x00079242</coc_coinsid>
        </co_configRow>
        <co_configRow>
          <kco>2</kco>
          <coc_name>COINS Limited</coc_name>
          <coc_lastrev>2013-02-02</coc_lastrev>
          <coc_vatregno>578524893</coc_vatregno>
          <coc_add1>The COINS Building</coc_add1>
          <coc_add2>12 The Grove</coc_add2>
          <coc_add3>Slough</coc_add3>
          <coc_add4>Berkshire</coc_add4>
          <coc_pcode>SL1 1QP</coc_pcode>
          <coc_phone>01753 711000</coc_phone>
          <coc_fax>01753 711010</coc_fax>
          <coc_reg/>
          <cnt_code/>
          <cur_code>GBP</cur_code>
          <coc_contref>607 1440 9</coc_contref>
          <coc_coinsid>2005-02-22T16:01:390x00079243</coc_coinsid>
        </co_configRow>
      </Body>
    </COINSInterfaceResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

We have now proved that:

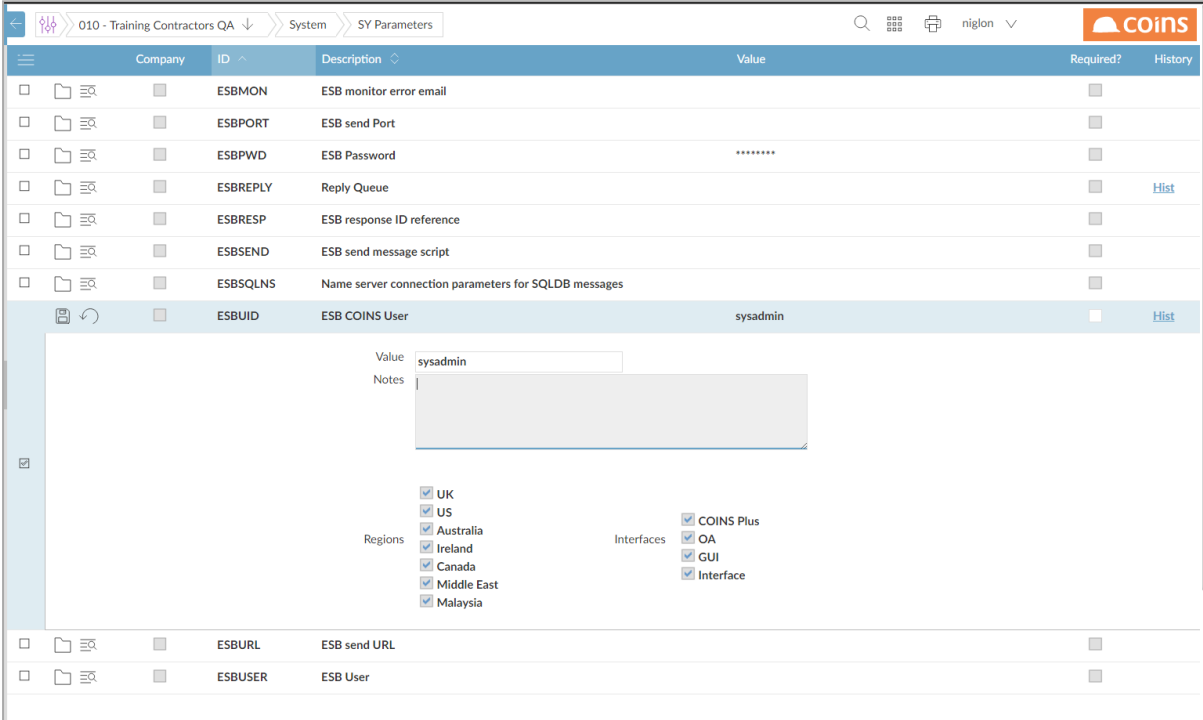
- the Services are working
- we can configure the messages appropriately
- we can call them and get a response

Having established these points, we can confidently move onto other aspects of using the services knowing that we have a properly configured and working system.

## 9.5 ESB COINS User

In the previous section, we had to specify a User and Password in the message in order to obtain a correct response.

Depending on your requirements, you may wish to have a single user account dedicated to running services. To configure this, Navigate to System/System Setup/System Parameters.



The screenshot shows the 'SY Parameters' configuration page in the COINS system. The table below lists the parameters, with 'ESBUID' selected for editing.

Company	ID	Description	Value	Required?	History
	ESBMON	ESB monitor error email		<input type="checkbox"/>	
	ESBPORT	ESB send Port		<input type="checkbox"/>	
	ESBPWD	ESB Password	*****	<input type="checkbox"/>	
	ESBREPLY	Reply Queue		<input type="checkbox"/>	<a href="#">Hist</a>
	ESBRESP	ESB response ID reference		<input type="checkbox"/>	
	ESBSEND	ESB send message script		<input type="checkbox"/>	
	ESBSQLNS	Name server connection parameters for SQLDB messages		<input type="checkbox"/>	
	ESBUID	ESB COINS User	sysadmin	<input type="checkbox"/>	<a href="#">Hist</a>

The 'ESBUID' configuration details are shown below the table:

- Value: sysadmin
- Notes: [Empty text area]
- Regions:
  - UK
  - US
  - Australia
  - Ireland
  - Canada
  - Middle East
  - Malaysia
- Interfaces:
  - COINS Plus
  - OA
  - GUI
  - Interface

Parameter ESBUID allows you to specify a user id (licenced to use Web Services) that will be used to run a service if no other userid and password are specified. Whilst we have used it in our example, it is recommended that you do NOT use sysadmin as the userid but instead set up an account dedicated for running web services.

In our previous example, if we now remove the <Login> section, containing our original username and password, from our message....

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <COINSInterface>
      <Header id="?" confirm="?" action="?" entity="?" arguments="?" ackID="?" testMsg="?">
        <!--Optional:-->
        <UserID?</UserID>
        <!--Optional:-->
        <From?</From>
        <!--Optional:-->
        <HostName>dev.coins-global.com</HostName>
        <!--Optional:-->
        <Environment>dev</Environment>
        <!--Optional:-->
        <Created?</Created>
        <!--Optional:-->
        <Login>
          <!--Optional:-->
          <User>niglon</User>
          <!--Optional:-->
          <Password>niglon</Password>
          <!--Optional:-->
          <CID?</CID>
        </Login>
      </Header>
      <Body>
        <!--Optional:-->
        <CompanyList>*</CompanyList>
        <!--Optional:-->
        <FromDate>01/01/12</FromDate>
      </Body>
    </COINSInterface>
  </soapenv:Body>
</soapenv:Envelope>
```

...and resubmit the request, the RESPONSE message is returned with the account from the system parameter returned as the authenticated user.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001/X
  <soapenv:Body>
    <COINSInterfaceResponse>
      <Header id="?" ackID="?" action="RESPONSE" entity="jcesb008">
        <UserID>SYSADMIN</UserID>
        <From>COINS</From>
        <HostName>dev.coins-global.com</HostName>
        <Environment>dev</Environment>
        <Created>2014-03-26T15:05:45.180+00:00</Created>
      </Header>
      <Body>
        <co_configRow>
          <kco>1</kco>
          <coc_name>COINS</coc_name>
          <coc_lastrev>2013-05-09</coc_lastrev>
          <coc_vatregno>578524893</coc_vatregno>
          <coc_add1>12 The COINS Budfilding</coc_add1>
          <coc_add2>12 Tdfhe Grove</coc_add2>
```

## 9.6 User Defined Services

Where a COINS Webservice does not exist that meets your requirements, User Defined Services, as their name implies, allow Web Services to be configured to access data to your specification.

User defined services come in several forms:

- Data Sets – These allow you full control of the information being retrieved from COINS
- Pages – Simple maintenance of data within COINS, such as hold codes on invoices.
- Programs – Calculation programs, such as those used in Workflow, which can be run from Web Services
- GET/SET – A service which allows you to GET information from any table and SET information in any table (subject to system security and Business Logic rules).

## 9.6.1 Datasets

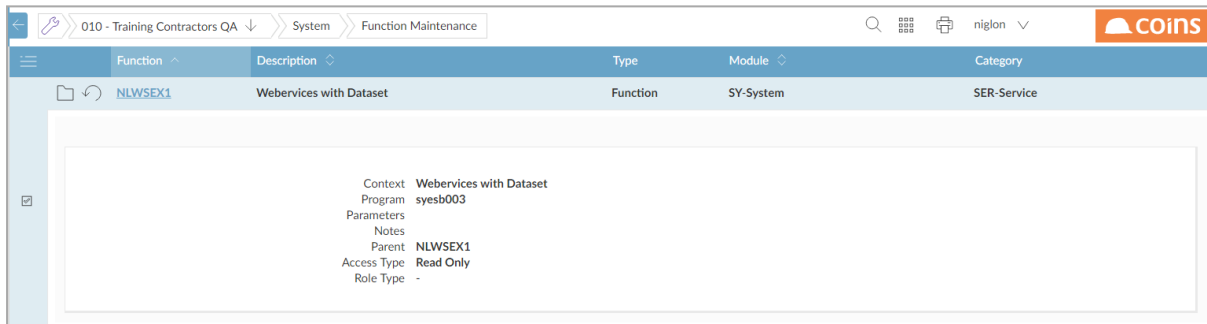
Three components are required for Datasets. A function, a page and the dataset itself.

It is important that all three components are defined with the same name.

In this example, we will use the name NLWSEX1 and we will create a dataset that will retrieve specified Supplier Records from the Supplier Master File.

### 9.6.1.1 Function

Create the function called NLWSEX1 with Category of SER – Service and Program of syesb003. The Context of the function will determine the description that will appear later on the Web Service menu.



### 9.6.1.2 Dataset

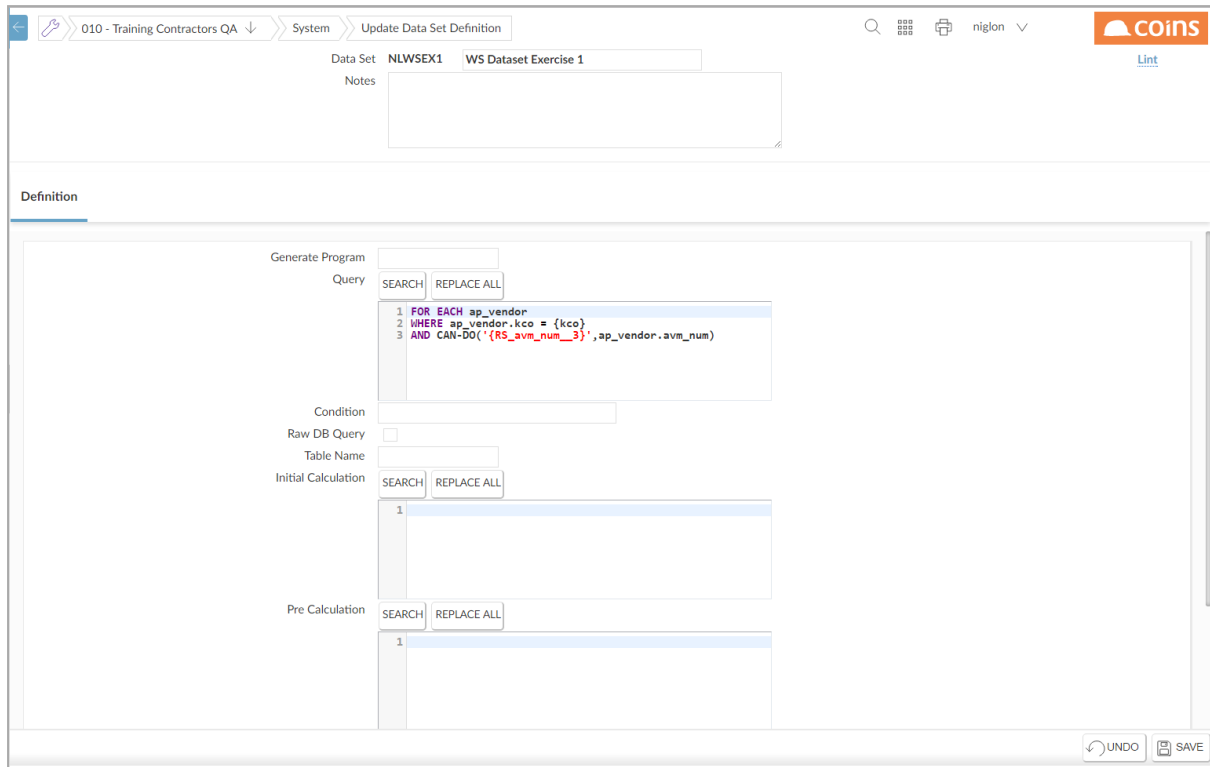
The dataset defines the output message of the webservice.

Create a dataset called NLWSEX1 with the Query:

```
FOR EACH ap_vendor  
WHERE ap_vendor.kco = {kco}  
AND CAN-DO('{RS_avm_num__3}',ap_vendor.avm_num)
```

This query will allow us to specify the COINS Company and a list of required Suppliers Accounts. These selections will be passed to the dataset by the page which we will create later.





Click

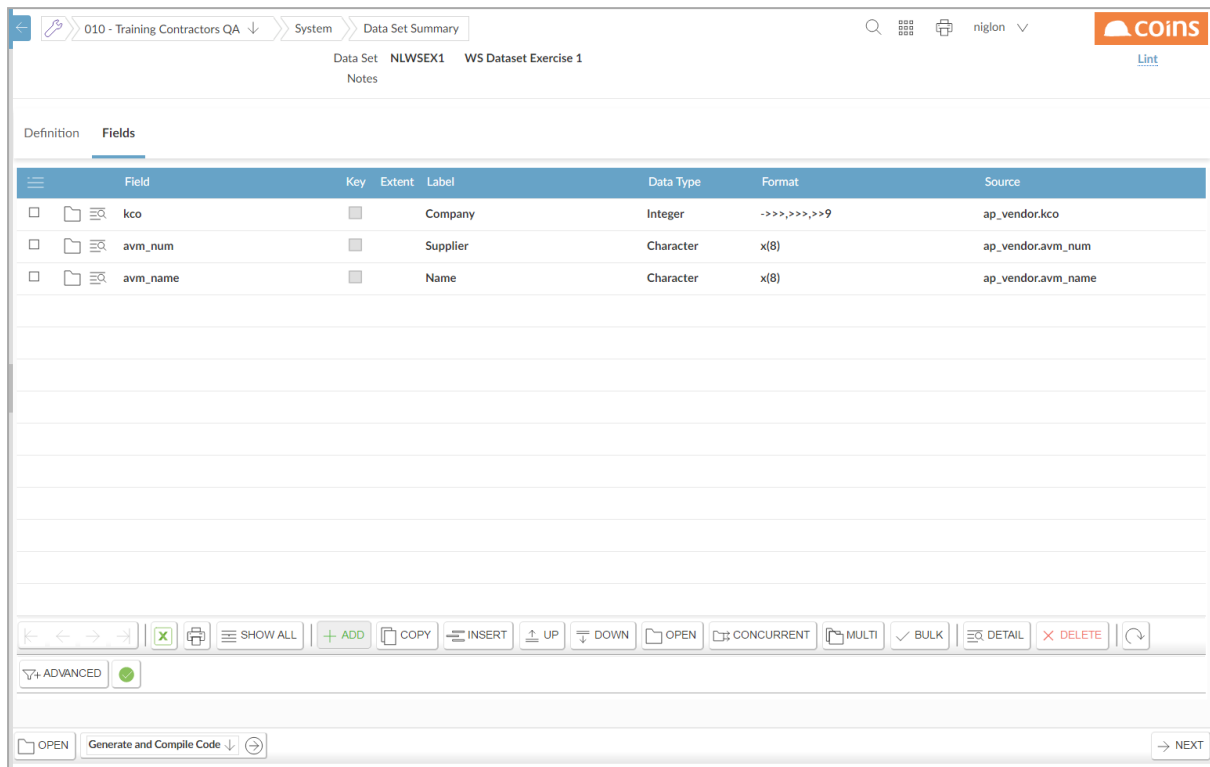
Add the fields

kco

avm\_num

avm\_name

The dataset is constructed in the same way as any OA Designer dataset, so fields can be calculated fields as well as taken directly from the COINS database.



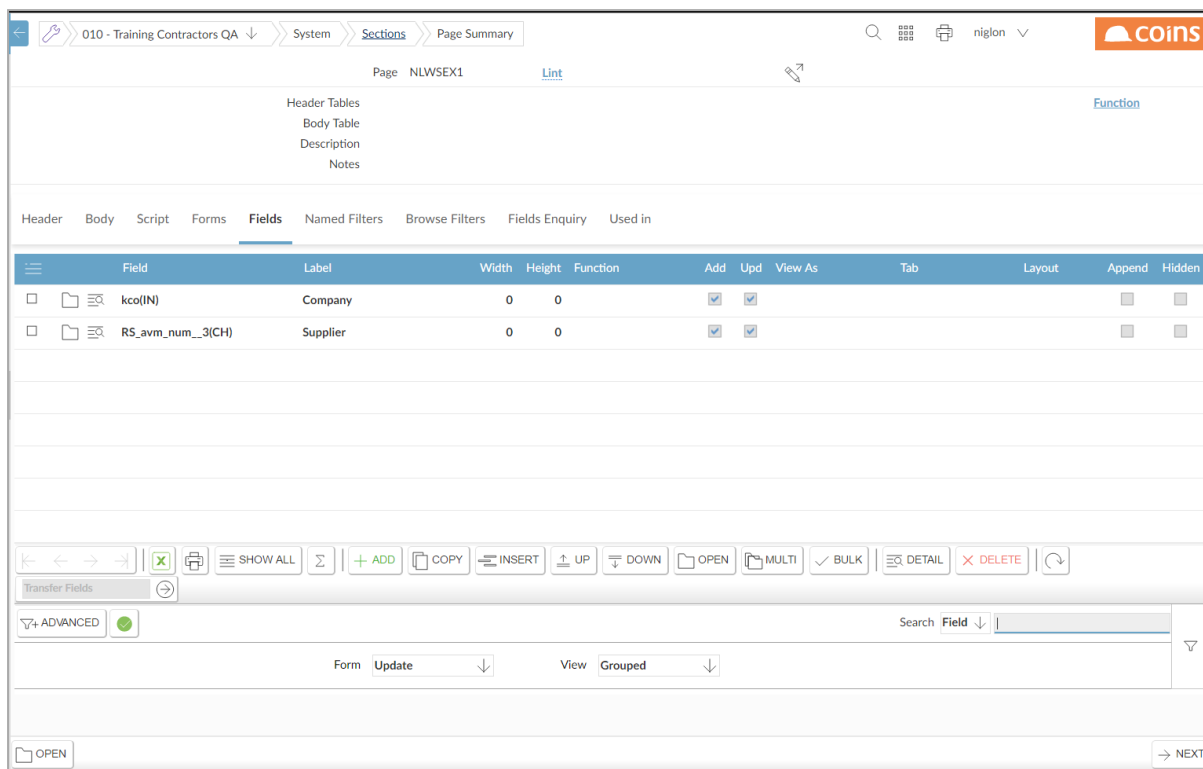
### 9.6.1.3 Page

The page design controls the fields that the web service will expect in order for it to perform its function.

In this example we need the web service to be passed the COINS Company (kco) and the list of required Supplier Accounts (RS\_avm\_num\_3).

For Web Services, the Page design consists only of a Page called the same name as the function and the Dataset, and the required fields defined on an UPDATE form.

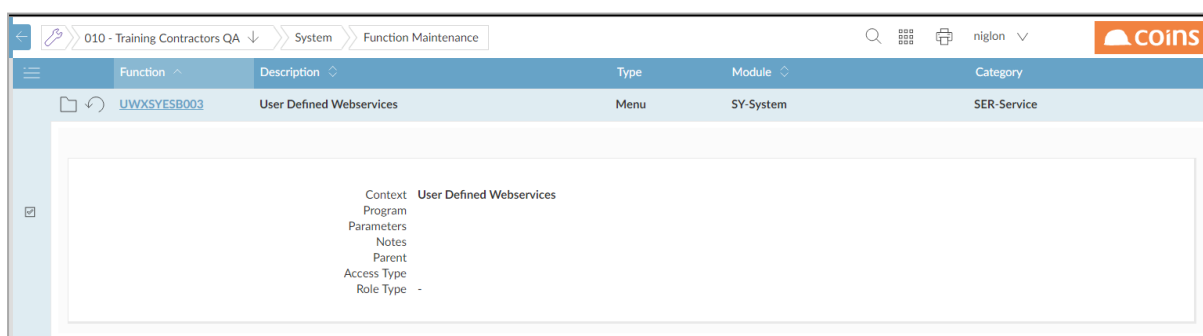
Field names can be followed by (XX) where XX is either IN,DE,DA,CH,LO meaning Integer, Decimal, Date, Character or Logical data type respectively. **If omitted then character data type is assumed.**



The page defines the INPUT schema for the BODY section of the message.

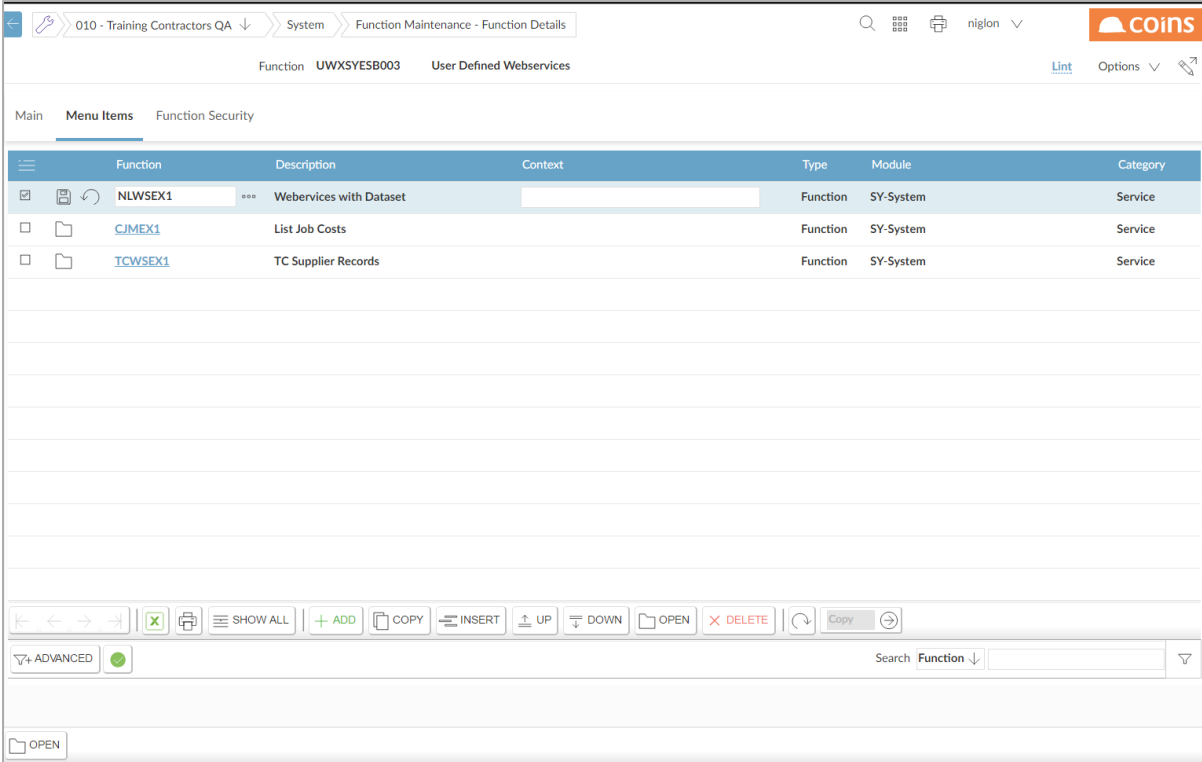
#### 9.6.1.4 Adding the Function to the Web services Menu

To run the Dataset Web Service, first check that the menu function UWXSYESB003 exists. If it does not exist, then you need to create it.



Open the Function and select the Menu Items Tab. If the function already exists, there may already be entries on this tab relating to Standard Used Defined Services issued by COINS (prefixed with %) or other User Defined Services created by your company.




Click and enter the name of your new User Defined Services Function (in our example NLWSEX1).



Function Maintenance - Function Details

Function: UWXSYESB003 User Defined Webservices


Main Menu Items Function Security

Function	Description	Context	Type	Module	Category
<input checked="" type="checkbox"/>  <a href="#">NLWSEX1</a> ***	Webervices with Dataset		Function	SY-System	Service
<input type="checkbox"/>  <a href="#">CJMEX1</a>	List Job Costs		Function	SY-System	Service
<input type="checkbox"/>  <a href="#">TCWSEX1</a>	TC Supplier Records		Function	SY-System	Service

Navigation: SHOW ALL, + ADD, COPY, INSERT, UP, DOWN, OPEN, DELETE, Copy

Search: Function ↓

OPEN

Click  . The full details of the function are displayed.

Navigate back to the COINS Services Menu and select the SY – System Hyperlink.

## COINS Services

Service	Description
CB	<a href="#">Cash Book</a>
EC	<a href="#">eCommerce</a>
FM	<a href="#">Facilities Management</a>
GL	<a href="#">General Ledger</a>
HS	<a href="#">House Sales</a>
HR	<a href="#">Human Resources</a>
JC	<a href="#">Contract Status</a>
PL	<a href="#">Purchase Ledger</a>
SC	<a href="#">Subcontract Ledger</a>
SL	<a href="#">Sales Ledger</a>
CS	<a href="#">Contract Sales Ledger</a>
SY	<a href="#">System</a>
VF	<a href="#">VAR</a>
PC	<a href="#">Plant</a>
DM	<a href="#">Document Management</a>
PO	<a href="#">Procurement</a>
CI	<a href="#">Company Information Workbench</a>
CC	<a href="#">Customer Care</a>
LA	<a href="#">Land Appraisal</a>
CV	<a href="#">CVR</a>
MK	<a href="#">Marketing</a>
MB	<a href="#">Mobile Applications</a>
SE	<a href="#">SE</a>
ST	<a href="#">Stock</a>

Locate the SYESB003 entries. Your new Function will be in this group.

# COINS Services

[COINS Services](#) >System

Service	Description
SYESB000	<a href="#">Confirmation Message</a>
SYESB001	<a href="#">Login</a>
SYESB002	<a href="#">Get Waiting Message Numbers</a>
SYESB003	<a href="#">Coxies ESB Service 001</a>
SYESB003	<a href="#">GLBAL Dataset</a>
SYESB003	<a href="#">JTS Data Set</a>
SYESB003	<a href="#">Activities Data Set</a>
SYESB003	<a href="#">Get jc_emprate for employee</a>
SYESB003	<a href="#">BIM Projects (All)</a>
SYESB003	<a href="#">BIM Projects (One)</a>
SYESB003	<a href="#">BIM PM Issue - Type</a>
SYESB003	<a href="#">BIM PM Issue - Impact</a>
SYESB003	<a href="#">BIM PM Issue - Importance</a>
SYESB003	<a href="#">BIM PM Issue - Status</a>
SYESB003	<a href="#">BIM PM Issues (Navis Views)</a>
SYESB003	<a href="#">BIM PM Project Team Personnel</a>
SYESB003	<a href="#">BIM PM Project Team Contacts</a>
SYESB003	<a href="#">BIM PM RFIs (Navis Views)</a>
SYESB003	<a href="#">BIM PM RFI - Status</a>
SYESB003	<a href="#">BIM PM RFI - Type</a>
SYESB003	<a href="#">BIM PM Issues</a>
SYESB003	<a href="#">BIM PM RFIs</a>
SYESB003	<a href="#">Coxies Designer Webservice Dataset</a>
SYESB003	<a href="#">Supplier Records</a>
SYESB004	<a href="#">My Page</a>
SYESB004	<a href="#">Timesheet Operator</a>



Select the Hyperlink to view the service schema.



**COINS Services**

COINS Services >System >Generic Dataset

<b>Service</b>	SYESB003 (NLWSEX1)
<b>Description</b>	Generic Dataset
<b>Schema</b>	<a href="#">input.xsd</a> <a href="#">output.xsd</a> <a href="#">exception.xsd</a> <a href="#">acknowledgement.xsd</a> <a href="#">SOAPinput.xsd</a> <a href="#">SOAPoutput.xsd</a>
<b>WSDL</b>	<a href="#">SYESB003.wsdl</a>
<b>WSDL NS</b>	<a href="#">SYESB003.wsdl</a>

The generic dataset service allows a designer to build input parameters to a dataset and request the data from the dataset using the service.  
The argument to this service is the function name of the function/page/dataset to be called.

**Input**

Entity	Type	Documentation
COINSInterface		
Header		
id	string	A unique message identifier from the originating system.
confirm	string	Whether a confirmation message is required. Set to true or yes to have a confirmation message returned. If this is set then an id must also be provided.
action	string	An action type that will indicate why the message was created. For example, this might be CREATE, UPDATE, or DELETE for a message as a result of a maintenance of a record or PUBLISH for the publish of some information.
entity	string	The service that should consume the message.
arguments	string	Arguments that should be passed to the service.

The Input Schema defined by the page will look something like this:

Sample

```
<COINSInterface>
<Header id="" confirm="" action="" entity="" arguments="" ackID="" testMsg="true">
  <UserID></UserID>
  <From></From>
  <HostName></HostName>
  <Environment></Environment>
  <Created>2014-03-28T09:05:40.967+00:00</Created>
  <Login>
    <User></User>
    <Password></Password>
  </Login>
  <CID>1</CID>
</Header>
</COINSInterface>
```

```
<extUser></extUser>  
  
<extAuth></extAuth>  
  
</Login>  
  
</Header>  
  
<Body>  
  
<Results>1</Results>  
  
<kco>99</kco>  
  
<RS_avm_num__3></RS_avm_num__3>  
  
</Body>  
  
</COINSInterface>
```

And the Output Schema, defined by the Dataset will look like this:

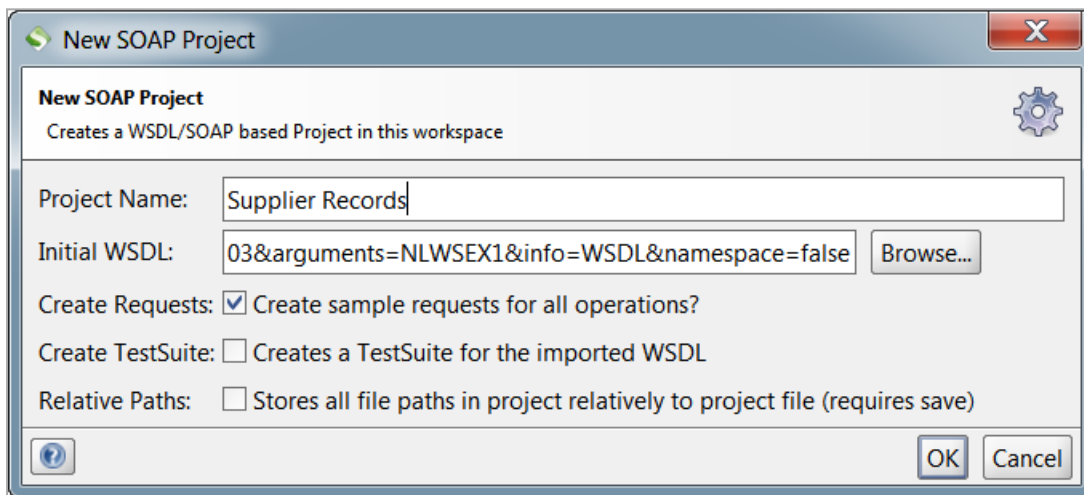
Sample

```
<COINSInterface>  
  
<Header id="" confirm="" action="" entity="" ackID="">  
  
<UserID></UserID>  
  
<From></From>  
  
<HostName></HostName>  
  
<Environment></Environment>  
  
<Created>2014-03-28T09:05:41.194+00:00</Created>  
  
</Header>  
  
<Body>  
  
<ttRow>  
  
<kco>99</kco>
```



```
<avm_num></avm_num>  
  
<avm_name></avm_name>  
  
</ttRow>  
  
</Body>  
  
</COINSInterface>
```

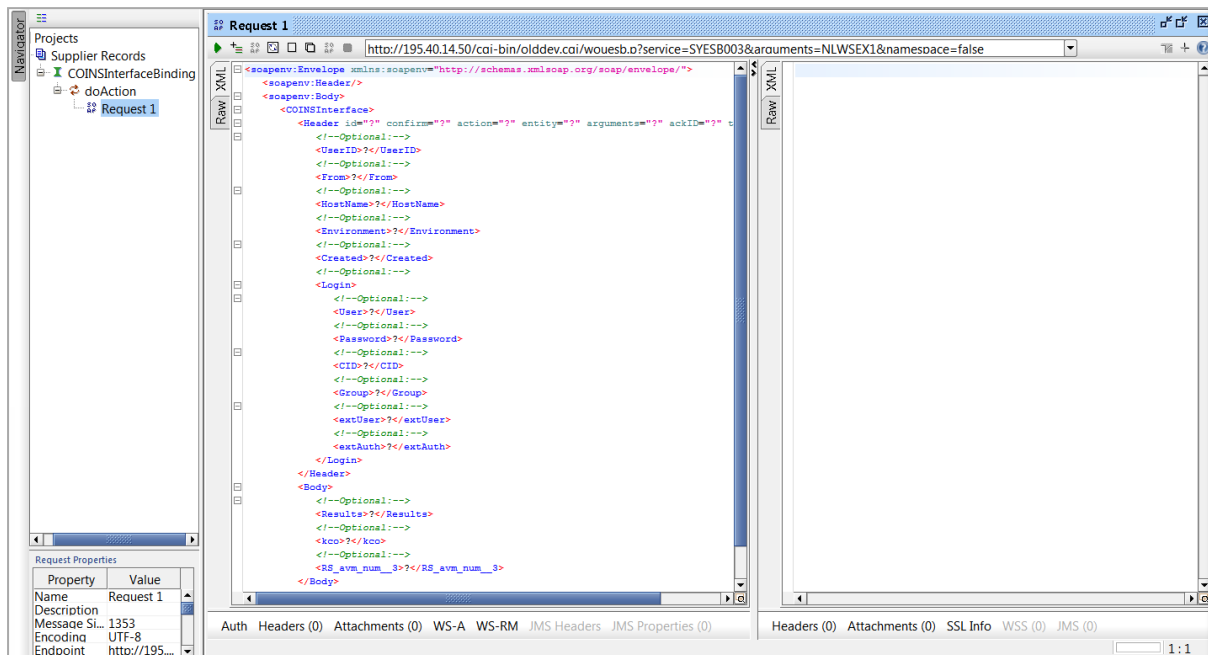
#### 9.6.1.5 Testing the Service



Create a new soapUI project and from the Service Schema, copy the WSDL shortcut link into the Initial WSDL field

Click OK.

Once the Project has been created. Run Request 1 in the editor (Double-click the Request 1 entry)



As before, we can use the input message as it is. However, this time we are going to replace the input message with the one generated in the Service Schema as it is a simpler form. Clear all the text in the left hand panel.

From the Service schema, scroll to the sample input message:

<input type="text" value="RS_avm_num_3"/>	<input type="text" value="string"/>	
---	-------------------------------------	--

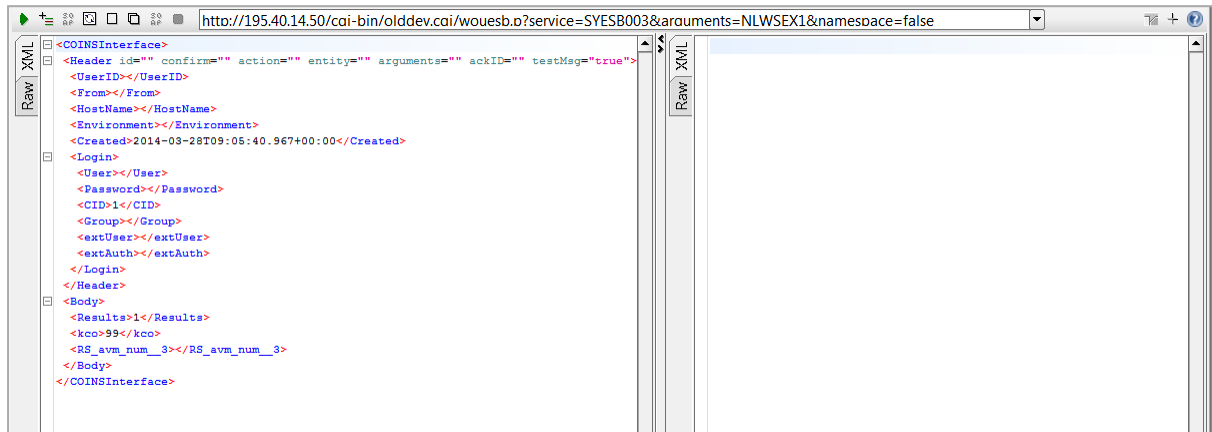
**Sample**

```
<COINSInterface>
<Header id="" confirm="" action="" entity="" arguments="" ackID="" testMsg="true">
  <UserID></UserID>
  <From></From>
  <HostName></HostName>
  <Environment></Environment>
  <Created>2014-03-28T09:05:40.967+00:00</Created>
  <Login>
    <User></User>
    <Password></Password>
    <CID>1</CID>
    <Group></Group>
    <extUser></extUser>
    <extAuth></extAuth>
  </Login>
</Header>
<Body>
  <Results>1</Results>
  <kco>99</kco>
  <RS_avm_num_3></RS_avm_num_3>
</Body>
</COINSInterface>
```

**Output**

Entity	Type	
COINSInterface		

Copy the text from <COINSInterface> to </COINSInterface> and paste into the soapUI right hand panel.



Enter the hostname, Environment and username/password detail.

Delete the lines for Group, extUser and extAuth

In the <Body> section, specify how many results are to be returned, the Company number to retrieve the records from and a comma separated list of Supplier Account numbers to retrieve (or \* for all). For example:

```
<COINSInterface>
<Header id="" confirm="" action="" entity="" arguments="" ackID="" testMsg=""
<UserID></UserID>
<From></From>
<HostName>dev.coins-global.com</HostName>
<Environment>dev</Environment>
<Created>2014-03-28T09:05:40.967+00:00</Created>
<Login>
<User>niglon</User>
<Password>niglon</Password>
<CID>1</CID>
</Login>
</Header>
<Body>
<Results>10</Results>
<kco>1</kco>
<RS_avm_num_3>*</RS_avm_num_3>
</Body>
</COINSInterface>
```

Send the message. Correct any exceptions if they occur.

If all is correct, a RESPONSE message should be returned along with the requested records:



```
<COINSInterface>
  <Header action="RESPONSE" entity="syesb003">
    <UserID>NIGLON</UserID>
    <From>COINS</From>
    <HostName>dev.coins-global.com</HostName>
    <Environment>dev</Environment>
    <Created>2014-03-28T09:45:40.563+00:00</Created>
  </Header>
  <Body>
    <ttRow>
      <kco>1</kco>
      <avm_num>---002</avm_num>
      <avm_name>Callander Utilites (AleBak Tes</avm_name>
    </ttRow>
    <ttRow>
      <kco>1</kco>
      <avm_num>.CA001</avm_num>
      <avm_name>trete</avm_name>
    </ttRow>
    <ttRow>
      <kco>1</kco>
      <avm_num>.CA003</avm_num>
      <avm_name>Carter Inc 3</avm_name>
    </ttRow>
    <ttRow>
      <kco>1</kco>
      <avm_num>000001</avm_num>
      <avm_name>Andrews Sykes Patent Glazing</avm_name>
    </ttRow>
    <ttRow>
      <kco>1</kco>
      <avm_num>000002</avm_num>
      <avm_name>G I Neilsons</avm_name>
    </ttRow>
    <ttRow>
      <kco>1</kco>
      <avm_num>000003</avm_num>
      <avm_name>IARNROD EIREANN</avm_name>
    </ttRow>
  </Body>
</COINSInterface>
```

Each record contains the fields as defined by the dataset. To add additional fields, simply add them to the dataset definition and re-run the request. The Page definition only needs to be amended if additional selection criteria are needed in the Input message.

## 9.6.2 Pages

Within COINS, pages allow the maintenance of data through Add, Update and Delete functionality. This functionality can be replicated by Web Services.

The user-defined page maintenance webservice capability is intended to be used for straight-forward maintenance of setup tables etc. where there are relatively few fields or no fields with dependencies.

For areas such as supplier maintenance etc. there are hard-coded services for this type of functionality.

Before embarking on designing your own page maintenance webservice it is recommended you seek advice from COINS to check there is not already a service available that will fulfil your needs.

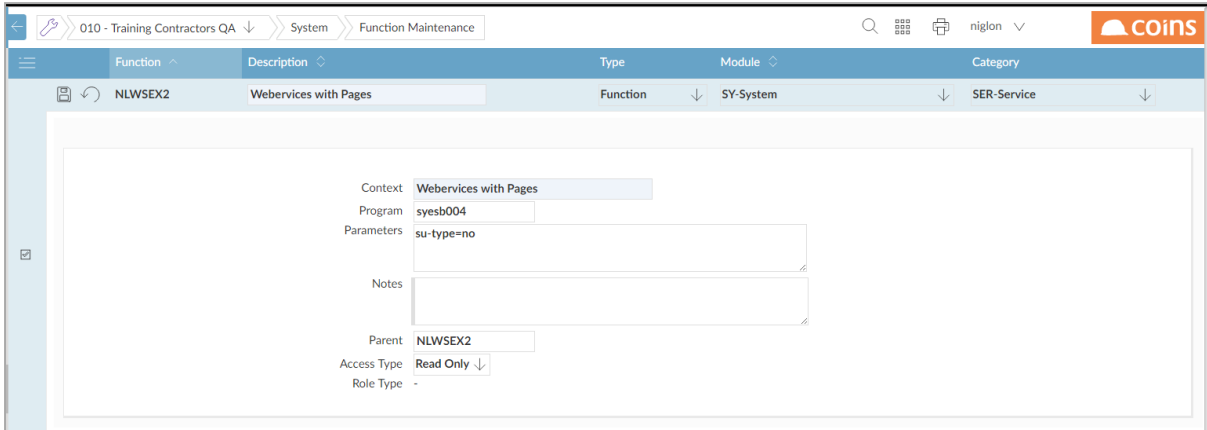
For a page webservice, two design components are required, the function(s) and a page. It is important that the function(s) and page have the same name.

In this example, we will use the name NLWSEX2 and we will design a page that will allow the maintenance of COINS UserID's.

### 9.6.2.1 Function

Create the function called NLWSEX2 with Category of SER – Service and program of syesb004. The Context of the function will determine the description that will appear later on the Web Services menu.

Since the table sysuser holds both groups and users, and we only want to access users for this example, we need to set a parameter on the function to ensure we maintain the correct records. Add the parameter su-type=no



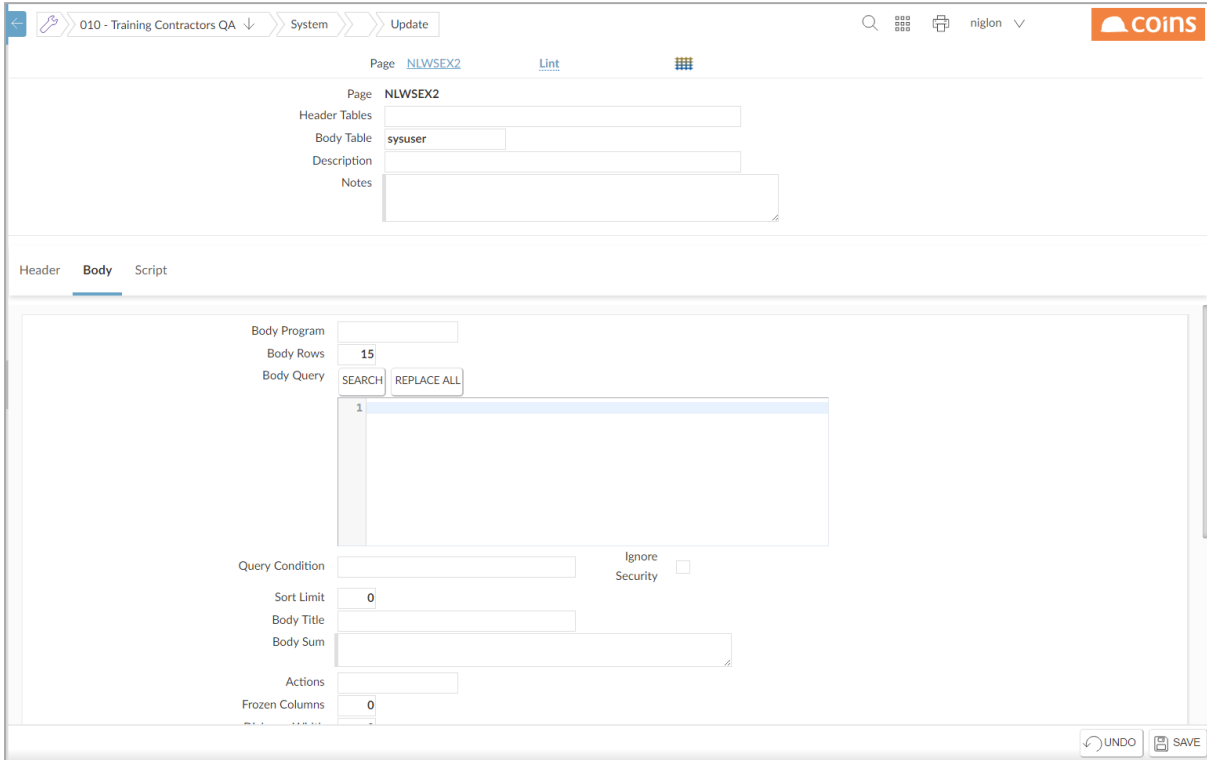
### 9.6.2.2 Page

The page controls the record access, the Input and the Output messages. These are specified using the following forms:

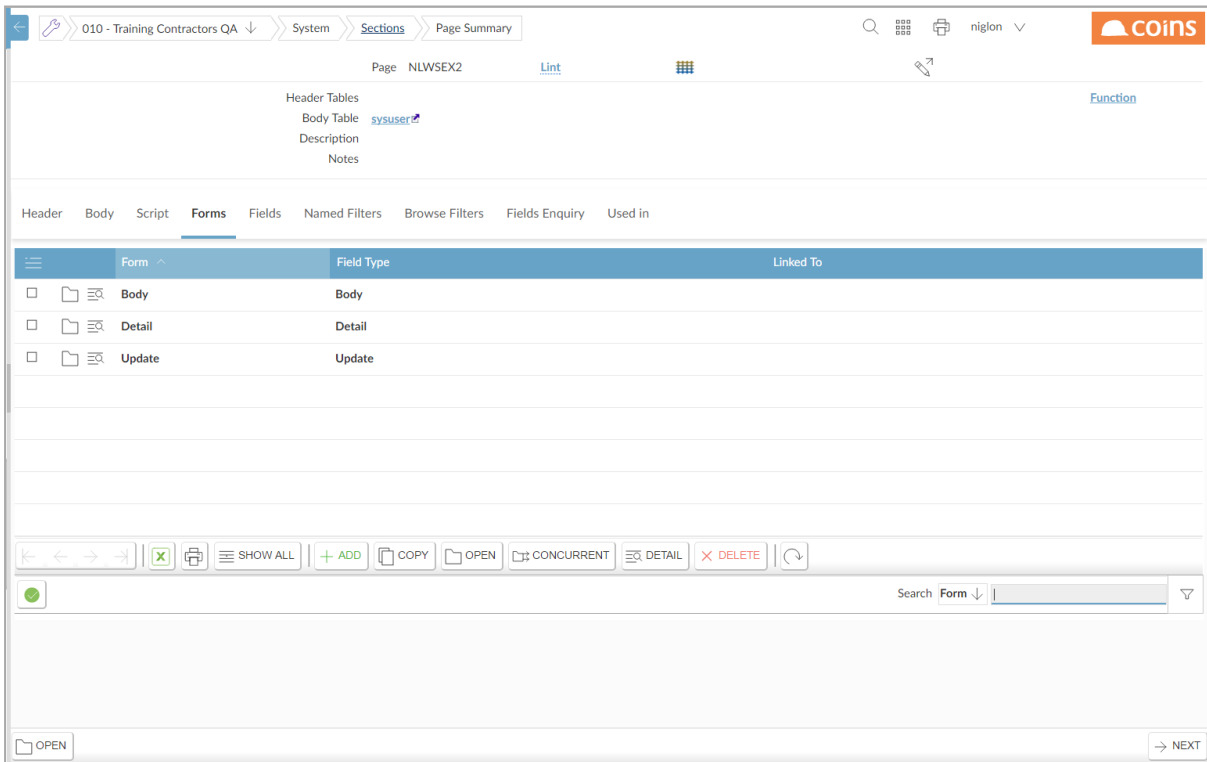
- Body        Defines the fields that uniquely identify the record
- Update     Defines the Input Schema
- Detail      Defines the output Schema

Create new page called NLWSEX2.

The body table should be set to the COINS database table that is to be maintained – in this example the table is sysuser.



### Add the forms Body, Detail and Update



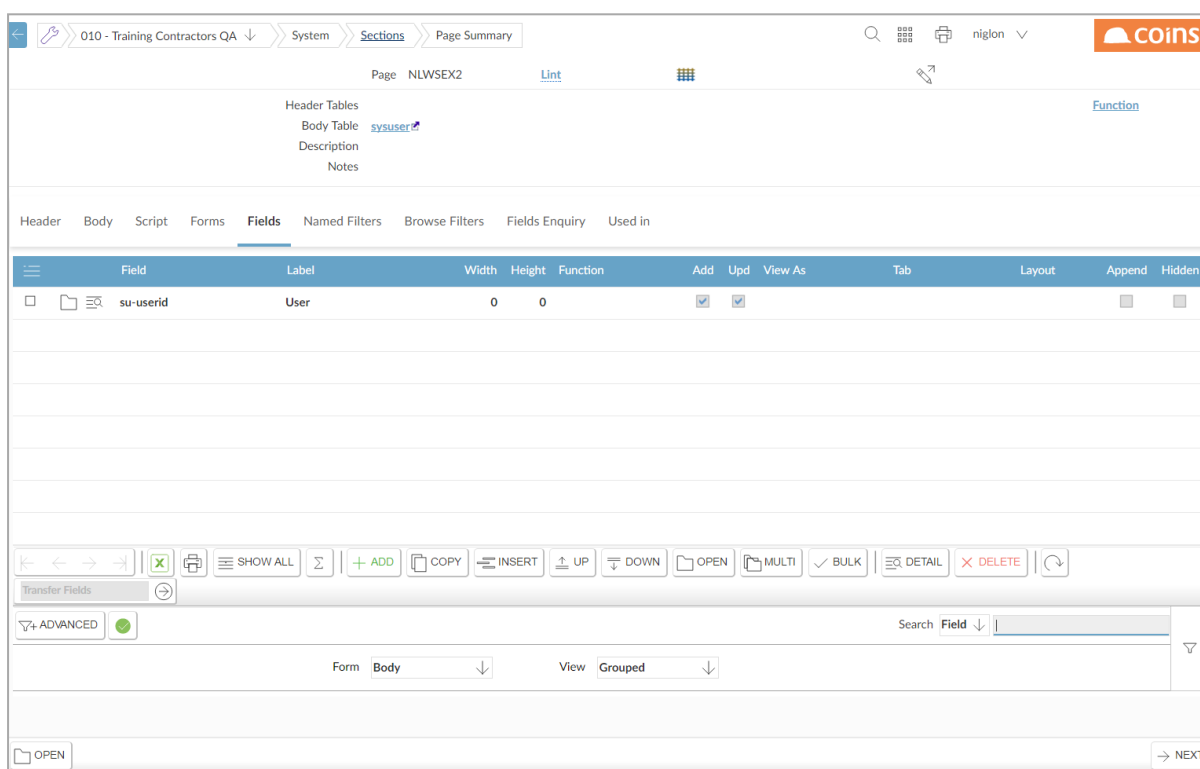
Form	Field Type	Linked To
Body	Body	
Detail	Detail	
Update	Update	

### Body Form

The Body Form is used to define which fields will be used to uniquely identify the record when Adding, Updating or Deleting. For sysuser the key field is the userid (su-userid). Other tables may have more than one field as part of the key – all should be added to ensure the correct record is accessed.

Add field su-userid to the Body Form.

Make sure that the Add and Update fields are ticked to indicate that this is a key field.



### Update Form

The update Form is used to define the Input Schema and should contain all the fields that will be maintained by the web service.

If records will be added by the Web service it is important to include all required fields (i.e. all fields that would be mandatory if added through a standard COINS page) as the business logic will not allow records to be Added without these fields.



For user records, the required fields are the User ID, the User Name and the Prime Company and a comma separated list of Companies the user can access. Our example web service will also maintain the Name User field. So our list of fields will be:

su-userid

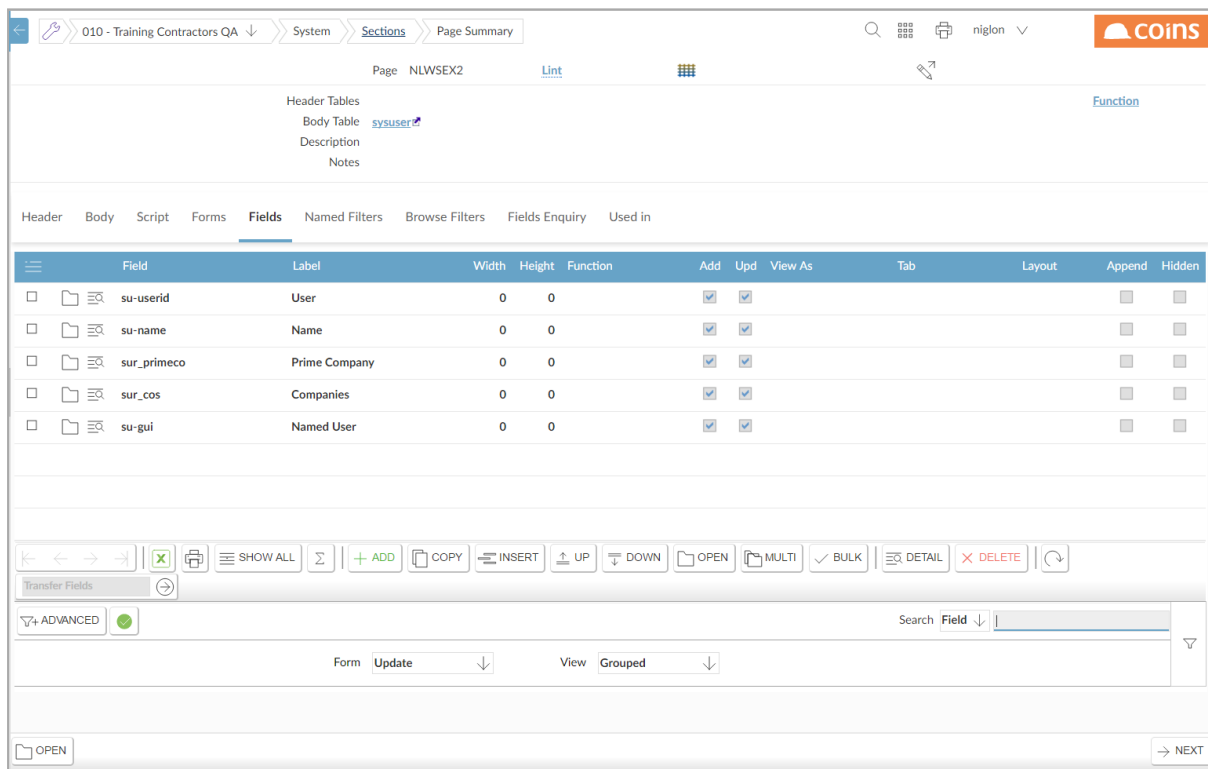
su-name

sur-primeco

su-cos

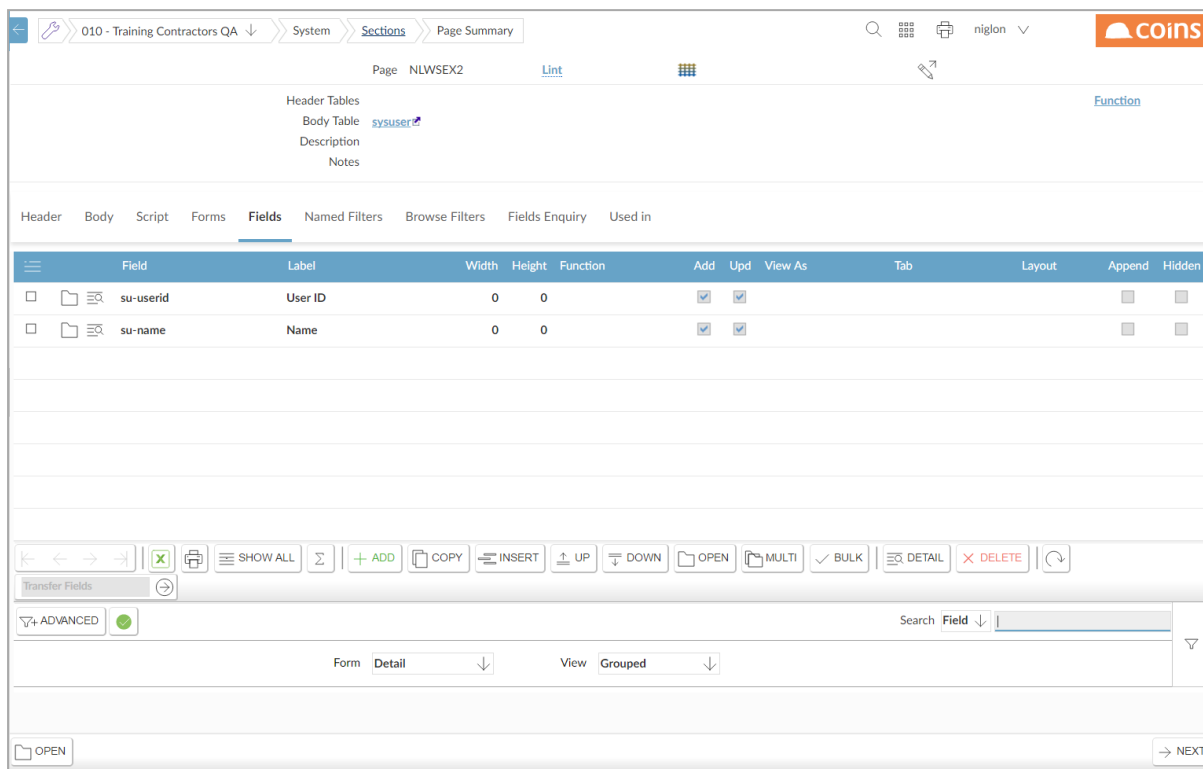
su-gui

Add these fields to the Update Form.



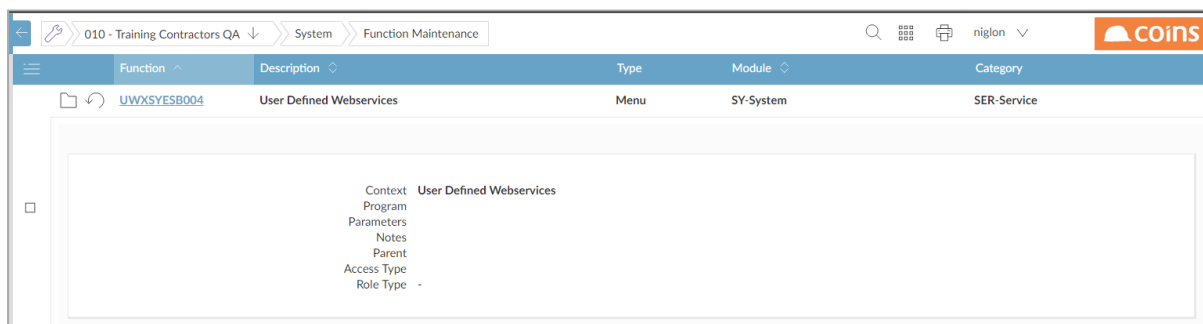
### Detail Form

The Detail Form defines the Output Schema and you may include any fields that you wish to see as confirmation of the record maintenance.



### 9.6.2.3 Adding the Function to the Web services Menu

To run the Page Web Service, first check that the menu function UWXSYESB004 exists. If it does not exist, then you need to create it.



Open the Function and select the Menu Items Tab. If the function already exists, there may already be entries on this tab relating to Standard Used Defined Services issued by COINS (prefixed with %) or other User Defined Services created by your company.

Click Add and enter the name of your new User Defined Services Function (in our example NLWSEX1).



010 - Training Contractors QA >> System >> Function Maintenance - Function Details

Function UWXSYESB004 User Defined Webservices

Main Menu Items Function Security

Function	Description	Context	Type	Module	Category
NLWSEX2	Webervices with Pages		Function	SY-System	Service

SEARCH Function ↓

Navigate back to the COINS Services Menu and select the SY – System Hyperlink.

## COINS Services

Service	Description
CB	<a href="#">Cash Book</a>
EC	<a href="#">eCommerce</a>
FM	<a href="#">Facilities Management</a>
GL	<a href="#">General Ledger</a>
HS	<a href="#">House Sales</a>
HR	<a href="#">Human Resources</a>
JC	<a href="#">Contract Status</a>
PL	<a href="#">Purchase Ledger</a>
SC	<a href="#">Subcontract Ledger</a>
SL	<a href="#">Sales Ledger</a>
CS	<a href="#">Contract Sales Ledger</a>
SY	<a href="#">System</a>
VF	<a href="#">VAR</a>
PC	<a href="#">Plant</a>
DM	<a href="#">Document Management</a>
PO	<a href="#">Procurement</a>
CI	<a href="#">Company Information Workbench</a>
CC	<a href="#">Customer Care</a>
LA	<a href="#">Land Appraisal</a>
CV	<a href="#">CVR</a>
MK	<a href="#">Marketing</a>
MB	<a href="#">Mobile Applications</a>
SE	<a href="#">SE</a>
ST	<a href="#">Stock</a>

Locate the SYESB00 entries. Your new Function will be in this group.

# COINS Services

[COINS Services](#) >System

Service	Description
SYESB000	<a href="#">Confirmation Message</a>
SYESB001	<a href="#">Login</a>
SYESB002	<a href="#">Get Waiting Message Numbers</a>
SYESB003	<a href="#">Webservices with Dataset</a>
SYESB004	<a href="#">Webservices with Pages</a>
SYESB006	<a href="#">Web service for Appointments/Tasks</a>
SYESB007	<a href="#">Field Get/Set</a>
SYESB011	<a href="#">Save Completed Mobile Form</a>
SYESB012	<a href="#">Database Publish Resend Data</a>
SYESB013	<a href="#">Update User Mobile Configuration</a>
SYESB014	<a href="#">Complete Workflow Stage</a>



Select the Hyperlink to view the service schema.



**COINS Services**

COINS Services > System > Generic Page

Service	SYESB004 (NLWSEX2)
Description	Generic Page
Schema	input.xsd output.xsd exception.xsd acknowledgement.xsd SOAPinput.xsd SOAPoutput.xsd
WSDL	SYESB004.wsdl
WSDL NS	SYESB004.wsdl

The generic page service allows a designer to build an OA page and call a bulk update using this service.  
The argument to this service is the function name of the page to be called.

**Input**

Entity	Type	Documentation
COINSInterface		
Header		
id	string	A unique message identifier from the originating system.
confirm	string	Whether a confirmation message is required. Set to true or yes to have a confirmation message returned. If this is set then an id must also be provided.
action	string	An action type that will indicate why the message was created. For example, this might be CREATE, UPDATE, or DELETE for a message as a result of a maintenance of a record or PUBLISH for the publish of some information.
entity	string	The service that should consume the message.
arguments	string	Arguments that should be passed to the service.
ackID	string	An optional acknowledgement ID. If this ID is set then COINS will respond with an acknowledgement message returning this ackID in the message header.
testMsg	boolean	Whether this message is a test message. A test message will process through in exactly the same way that a normal message would except that at the point where it would normally commit the transaction to the database, the message is then backed out.
UserID	string	The user in the originating system that caused the message to be produced.
From	string	The name of the system that produced the message.
HostName	string	This is the name of the host system (on UNIX the HOSTNAME environment variable). This is automatically checked (if present in the message) and if it does not match the host name of the system then an exception message is created. If the HostName tag is missing then HostName checking is omitted.
Environment	string	This is the name of the COINS environment that is the intended target for the message. This is automatically checked based on the name of the instance of the COINS system if it does not match then an exception message is created.
Created	dateTime	Date/Time of the source system when the message was created.
Login		
User	string	The COINS user to be used to consume the message. If omitted then the user set in SY parameter ESBUID will be used. If user is specified then a password must also be given.
Password		The password for the User specified above. The password can either be plain text or encoded to a 32 character hexadecimal string using an MD5 digest. For example the MD5 encoding for "The quick brown fox jumps over the lazy dog" is

The Input Schema – defined by the Update form of the page will look something like this:

**Sample**

```
<COINSInterface>
  <Header id="" confirm="" action="" entity="" arguments="" ackID="" testMsg="true">
    <UserID></UserID>
    <From></From>
    <HostName></HostName>
    <Environment></Environment>
    <Created>2014-04-08T14:02:17.967+01:00</Created>
    <Login>
      <User></User>
      <Password></Password>
      <CID>1</CID>
      <Group></Group>
      <extUser></extUser>
      <extAuth></extAuth>
    </Login>
  </Header>
  <Body>
    <sysuserRow rsp_action="" id="">
      <su-userid></su-userid>
      <su-name></su-name>
      <sur_primeco>99</sur_primeco>
      <sur_cos></sur_cos>
      <su-gui>true</su-gui>
    </sysuserRow>
  </Body>
</COINSInterface>
```

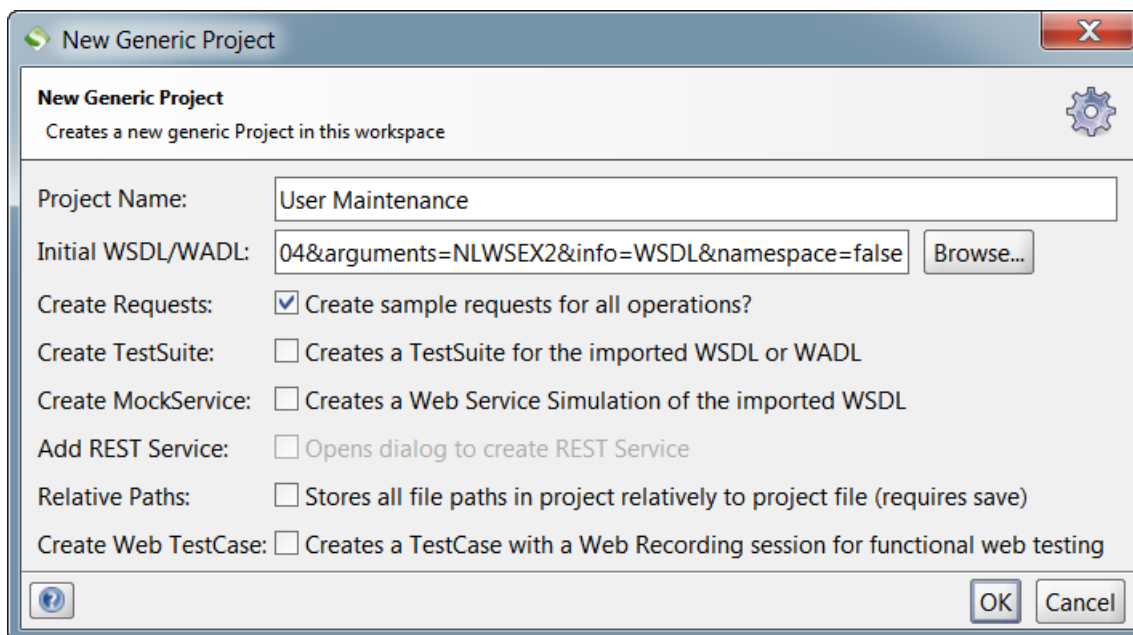
The Output Schema – defined by the Detail Form of the page will look something like this:

### Sample

```
<COINSInterface>
  <Header id="" confirm="" action="" entity="" ackID="">
    <UserID></UserID>
    <From></From>
    <HostName></HostName>
    <Environment></Environment>
    <Created>2014-04-08T14:02:18.055+01:00</Created>
  </Header>
  <Body>
    <sysuserRow rsp_action="" id="">
      <su-userid></su-userid>
      <su-name></su-name>
    </sysuserRow>
  </Body>
</COINSInterface>
```

---

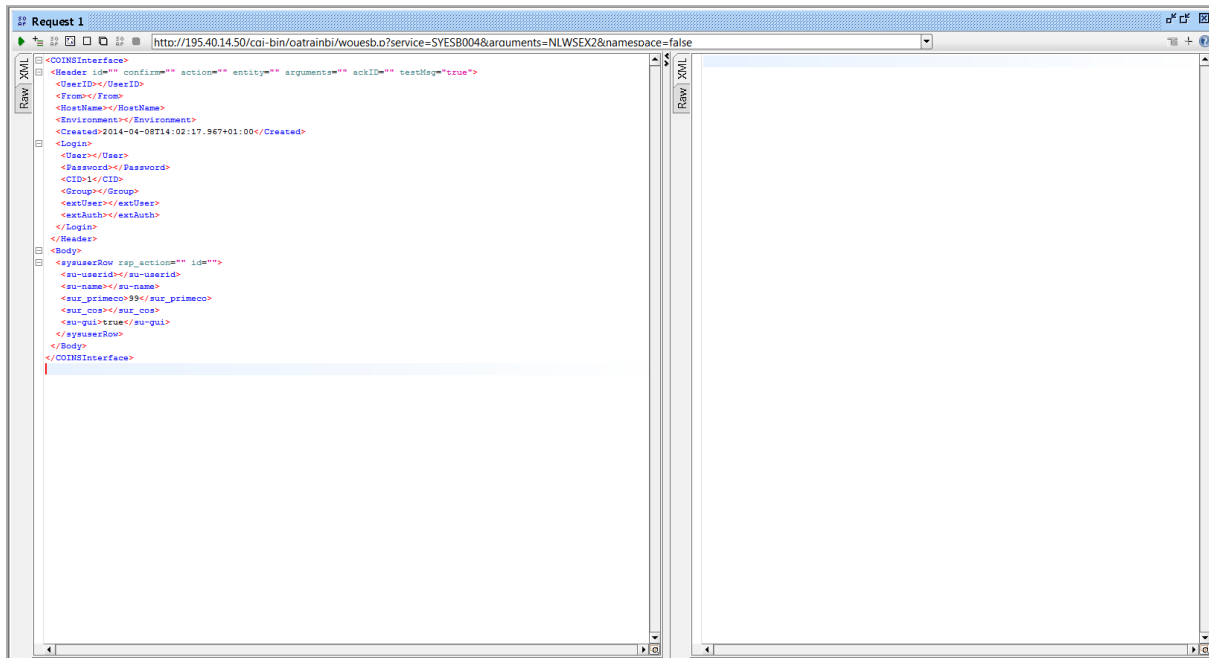
#### 9.6.2.4 Testing the Service



Create a new soapUI project and from the Service Schema copy the WSDL shortcut link and paste this into the Initial WSDL field

Click OK

Once the project has been created, run Request 1 in the editor. Replace the Input message with the simplified one from the Input Schema from the Web service.



Enter the hostname, Environment and username/password details.

Delete the lines for Group, extUser and extAuth

In the <Body> Section you will notice a new line

```
<sysuserRow rsp_action="" id="">
```

This “Row” record must contain a valid rsp\_action of one of the following:

- A Add
- U Update
- D Delete

The id field can contain an id record that will be useful for identifying issues when updating more than one record. We will use this later in the exercise.

Specify an rsp\_action of A and then fill in the appropriate user details into the remaining Body Fields.

For example:





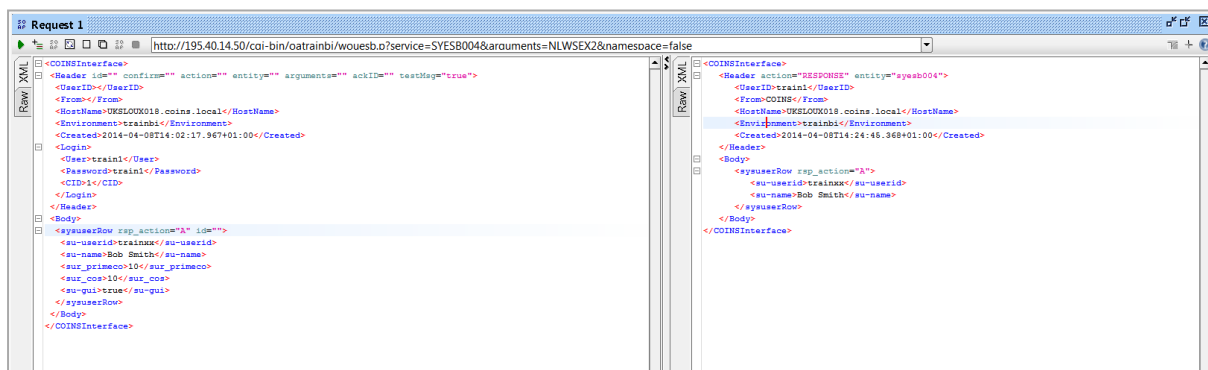
```

<COINSInterface>
  <Header id="" confirm="" action="" entity="" arguments="" ackID="" testMsg="true">
    <UserID></UserID>
    <From></From>
    <HostName>UKSLOUX018.coins.local</HostName>
    <Environment>trainbi</Environment>
    <Created>2014-04-08T14:02:17.967+01:00</Created>
  </Header>
  <Login>
    <User>train1</User>
    <Password>train1</Password>
    <CID>1</CID>
  </Login>
  </Header>
  <Body>
    <sysuserRow rsp_action="" id="">
      <su-userid>trainxx</su-userid>
      <su-name>Bob Smith</su-name>
      <sur_primeco>10</sur_primeco>
      <sur_cos>10</sur_cos>
      <su-gui>true</su-gui>
    </sysuserRow>
  </Body>
</COINSInterface>

```

Send the message and correct any exceptions if they occur.

If all is correct, a RESPONSE message should be returned.



However, if we check the User records in User Maintenance, no new User has been created.

If we look at the Input Message again, at the top of the message is another new line:

```

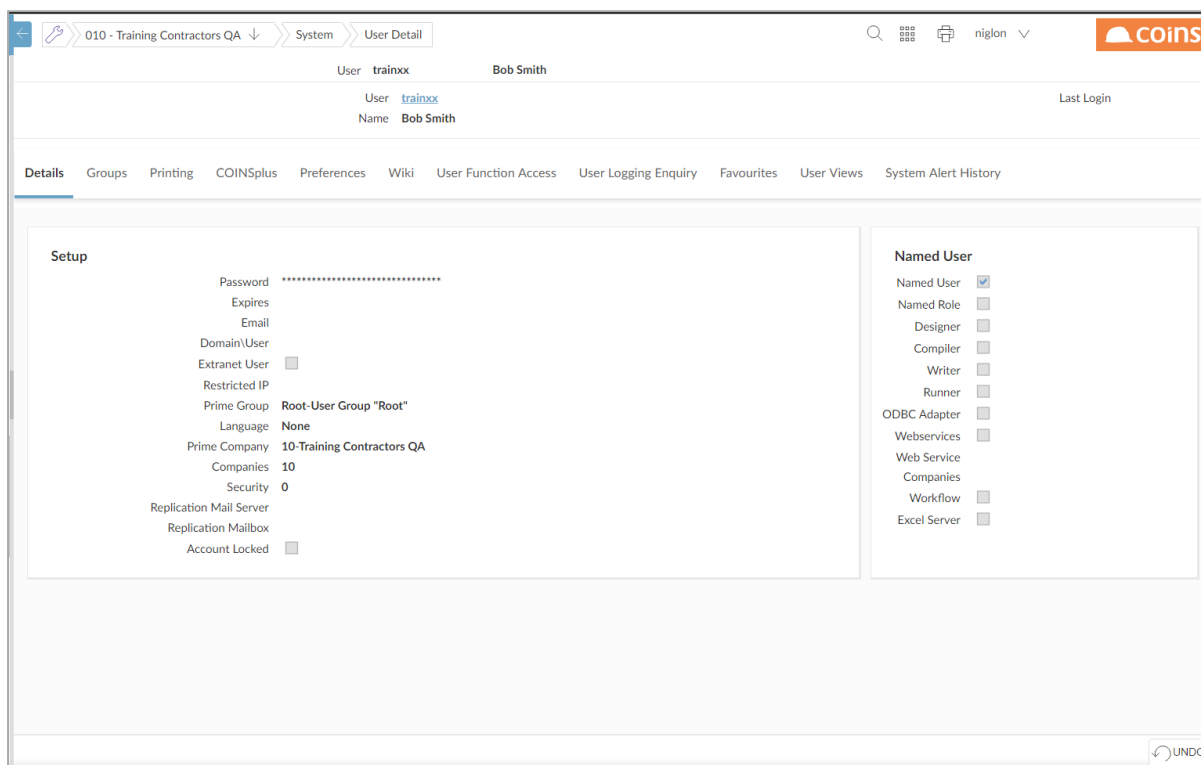
<COINSInterface>
  <Header id="" confirm="" action="" entity="" arguments="" ackID="" testMsg="true">

```

If this is set to “true”, the details will be sent and validated but no record maintenance will take place. This allows us to test the message without affecting any data. Only when this is set to “false” will record maintenance take effect.

Set this to false and send the message again.

This time, when we check user maintenance, we should see a new user record:



Experiment now with the `rsp_actions` of A, U and D to see the changes in the record and messages that appear in the output message. Try adding a record with the same userid for example or add a new user but leave the user name blank.

#### Using the ID field when adding multiple records

We can maintain more than one record at a time by duplicating the block of lines within the Body section of the Input Message. For example:

```
<Body>
  <sysuserRow rsp_action="A" id="record1">
    <su-userid>trainxx</su-userid>
    <su-name>Bob Smith</su-name>
    <sur_primeco>10</sur_primeco>
    <sur_cos>10</sur_cos>
    <su-gui>true</su-gui>
  <sysuserRow rsp_action="A" id="record2">
    <su-userid>trainxy</su-userid>
    <su-name>Barry Johns</su-name>
    <sur_primeco>10</sur_primeco>
    <sur_cos>10</sur_cos>
    <su-gui>true</su-gui>
  </sysuserRow>
</Body>
```

In the above example, in the id field of each block I have entered a unique reference for each record.

In the message I have specified an Add of two new records, but one of the userids already exists in the User records. When I then send the message one of the updates will fail. In the output message I will get an exception:

```
<COINSInterface>
  <Header action="EXCEPTION" entity="sysesb004">
    <UserID>train1</UserID>
    <From>COINS</From>
    <HostName>UKSLOUX018.coins.local</HostName>
    <Environment>trainbi</Environment>
    <Created>2014-04-08T14:37:11.304+01:00</Created>
  </Header>
  <Body>
    <Exception>
      <Exception>User already exists with that ID. [SY731] (Row:1 ID:record1)</Exception>
      <ThrownAt>preWrite sur-rsp.p,commit sur-rsp.p,commitrowUpdate sur-rsp.p,writeRecord sysesb004.p</ThrownAt>
    </Exception>
    <Input>
      <sysuserRow id="record1" rsp_action="A">
        <su-userid>trainxx</su-userid>
        <su-name>Bob Smith</su-name>
        <sur_primeco>10</sur_primeco>
        <sur_cos>10</sur_cos>
        <su-gui>true</su-gui>
      <sysuserRow id="record2" rsp_action="A">
        <su-userid>trainxy</su-userid>
        <su-name>Barry Johns</su-name>
        <sur_primeco>10</sur_primeco>
        <sur_cos>10</sur_cos>
        <su-gui>true</su-gui>
      </sysuserRow>
    </sysuserRow>
  </Input>
</Body>
</COINSInterface>
```

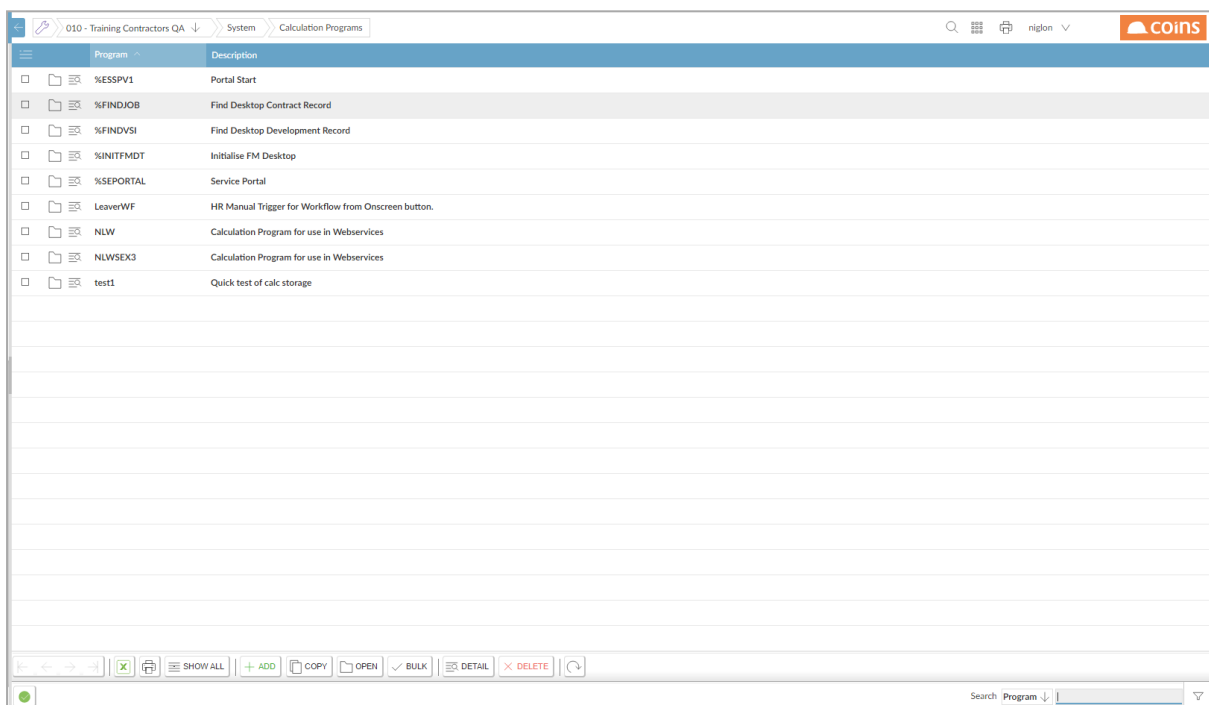
The exception will always give the row number that failed, but with an ID specified, the ID will also appear in the exception making it easier for me to identify the record that needs to be amended. Only when all records in the input message are correct will any changes be applied to the database.

### 9.6.3 Calculation Programs

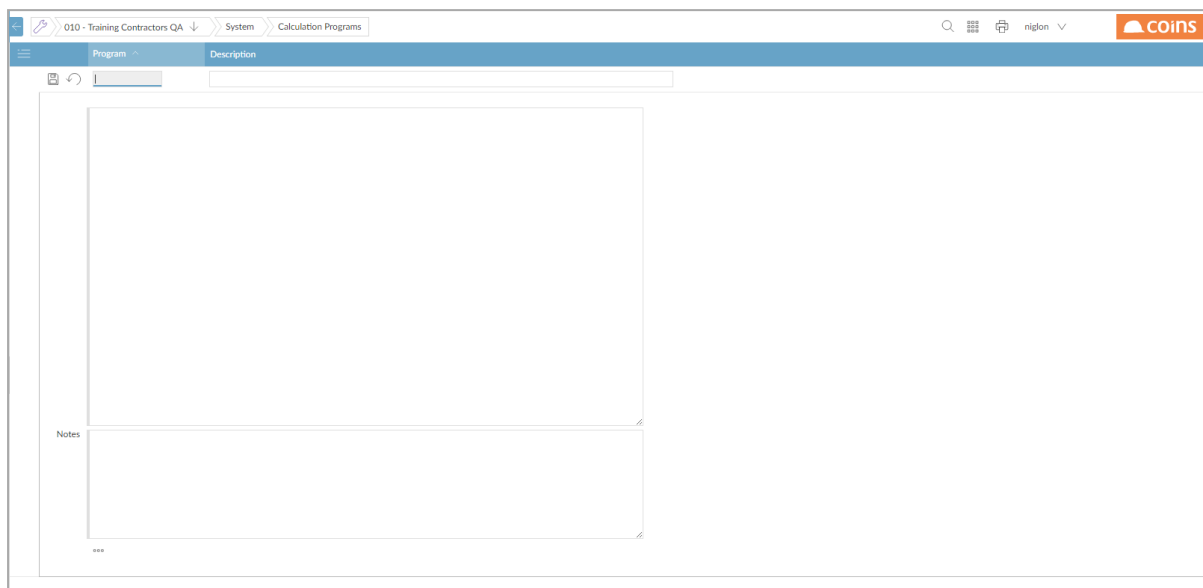
COINS provides the functionality for user-defined programs to be created and called by Web services offering greater flexibility for data manipulation and retrieval.

#### 9.6.3.1 Defining the Calculation

Calculation Programs may be defined using the Calculation Program option under OA Reporting & BI Setups in the OA Reporting & BI Module.



To create a new program, click



Program	This is the name by which the calculation will be referenced. It is recommended that this name is meaningful enough for its purpose to be easily inferred when debugging in the future.
Description	Give a brief description of the purpose of the calculation here.

The next box is for the calculation. The calculation syntax used follows COINS OA standard syntax, with numerical variables for the various parameters. Each numerical variable is encased in curly braces e.g. {1}

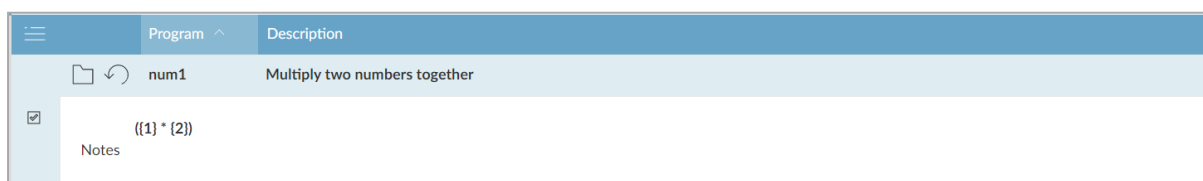
For example, if we wanted a calculation to multiply two different numbers, we would write this as:

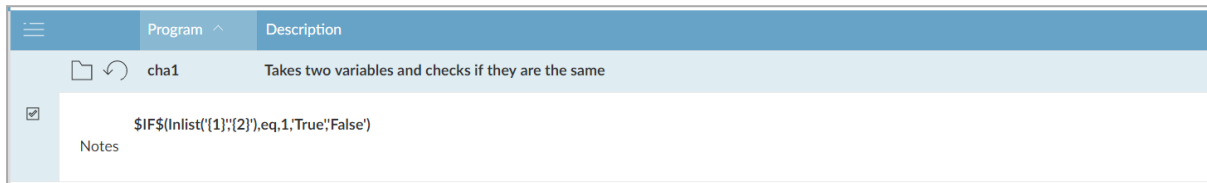
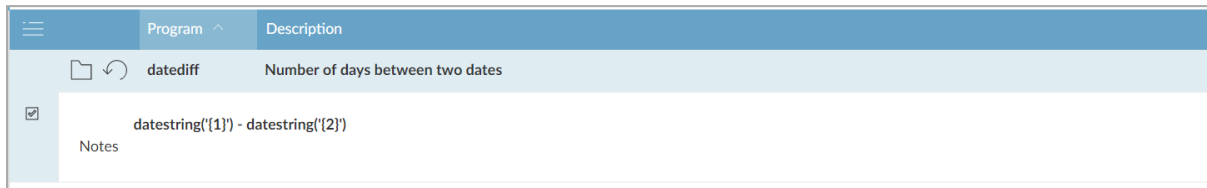
{1} \* {2}

Once a variable is defined, it can be re-used in a calculation and therefore need only be passed into the calculation once.

e.g. {1} \* {1} \* {1}

Three examples of a calculation program definition are shown below:





The calculation code can run other user defined calculation programs in the same way as any other calculation.

If the program is run without passing the required {} parameters then they will be replaced in the calculation with blanks.

Variables defined before the calculation is run can be accessed. Variables defined or updated in the calculation program will be available following the call to the calculation program.

The Notes section allows a more detailed description of the calculation to be entered.

#### 9.6.3.2 Testing Calculation Programs (run, run\$)

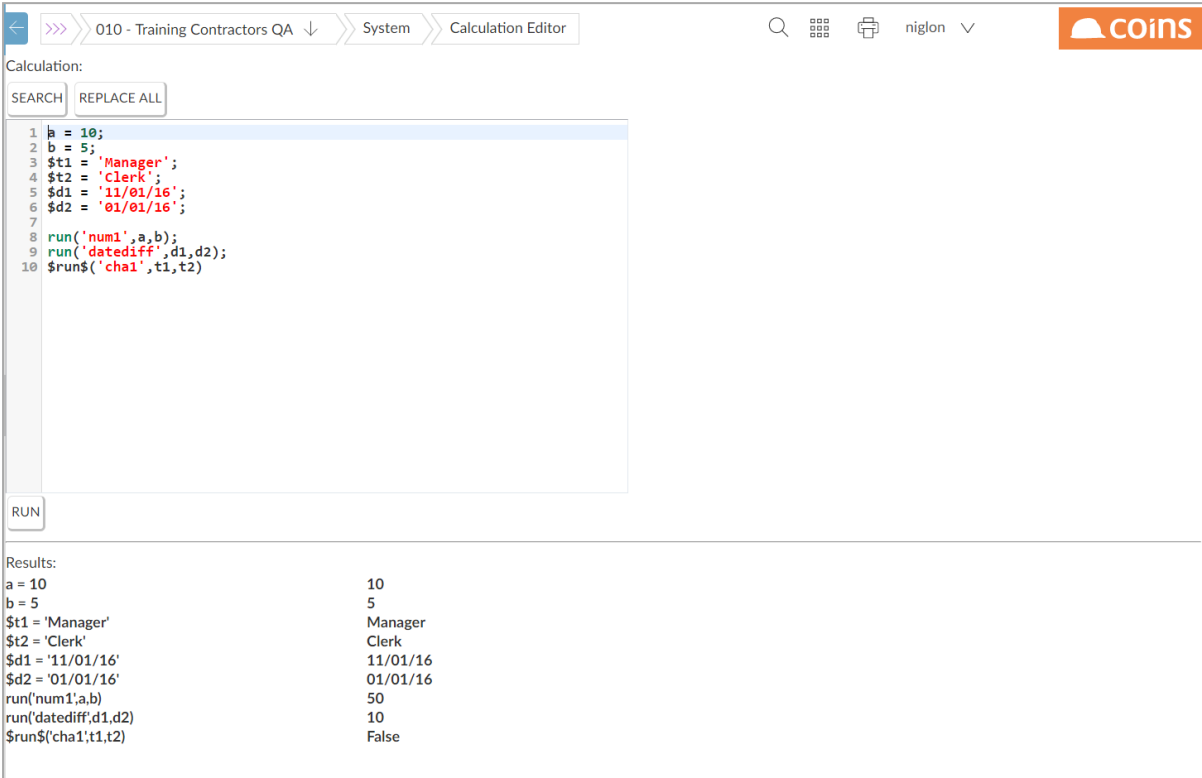
Once a calculation has been saved then it can be run by using:

**run('calc',var1,var2)** for numerical values

or

**\$run\$('calc',var1,var2)** for character values

Calculations may be tested in the Calculation editor prior to use to check they work as expected.



The screenshot shows the 'Calculation Editor' window in the COINS BI Toolset. The window title is '010 - Training Contractors QA' and it contains a 'System' tab and a 'Calculation Editor' tab. The editor contains the following code:

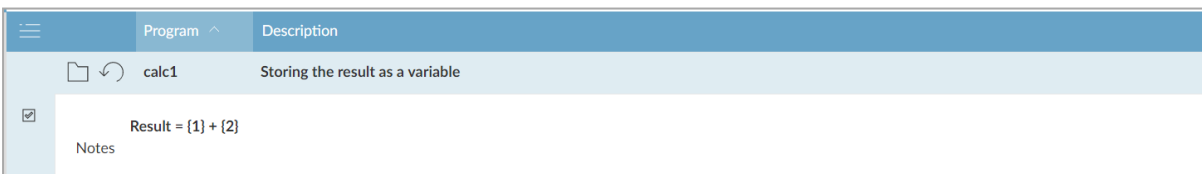
```

1  b = 10;
2  b = 5;
3  $t1 = 'Manager';
4  $t2 = 'Clerk';
5  $d1 = '11/01/16';
6  $d2 = '01/01/16';
7
8  run('num1',a,b);
9  run('datediff',d1,d2);
10 $run('cha1',t1,t2)
    
```

Below the code editor is a 'RUN' button. The 'Results' section displays the output of the calculation:

Results:	
a = 10	10
b = 5	5
\$t1 = 'Manager'	Manager
\$t2 = 'Clerk'	Clerk
\$d1 = '11/01/16'	11/01/16
\$d2 = '01/01/16'	01/01/16
run('num1',a,b)	50
run('datediff',d1,d2)	10
\$run('cha1',t1,t2)	False

Calculation programs can be defined to both use and create variables. For example:



The screenshot shows a table with two columns: 'Program' and 'Description'. The first row shows a program named 'calc1' with the description 'Storing the result as a variable'. Below this, a 'Notes' section contains the text 'Result = {1} + {2}'.

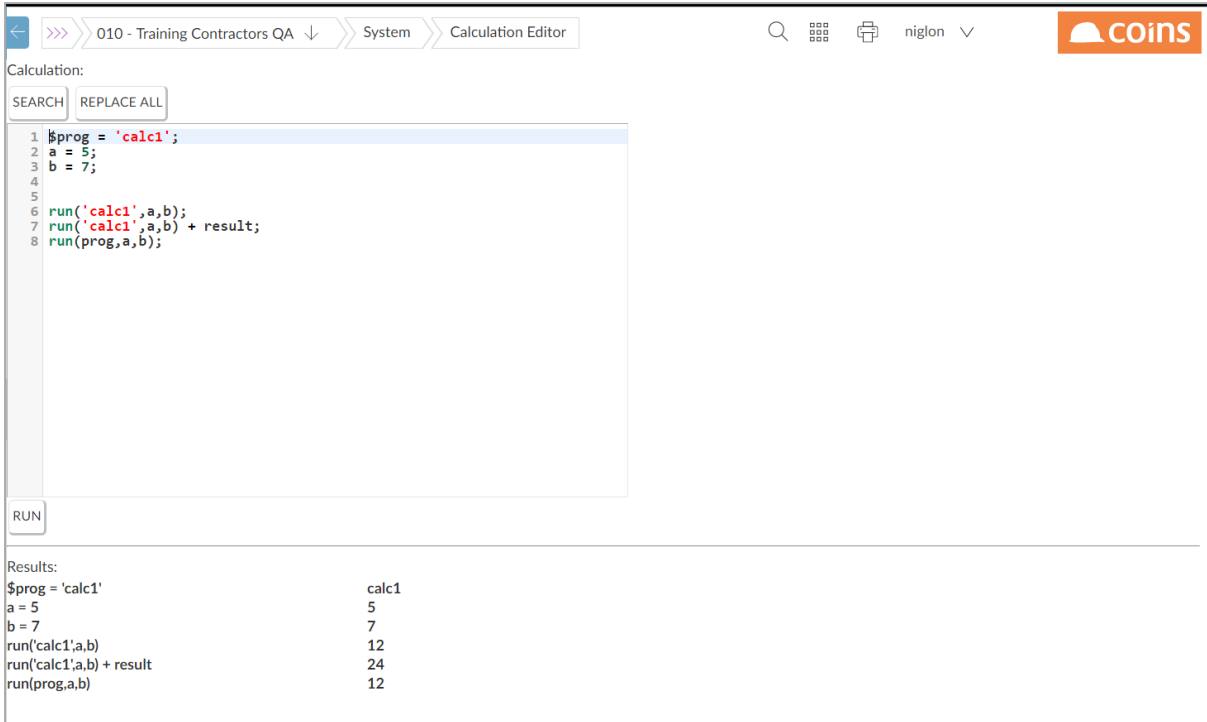
Program	Description
calc1	Storing the result as a variable

Notes: Result = {1} + {2}

This will store the value of {1} + {2} in a variable called result. This can then be used directly in subsequent calculations.

The name of the Calculation program to be called may also be passed as a variable.

Examples of these are shown below:



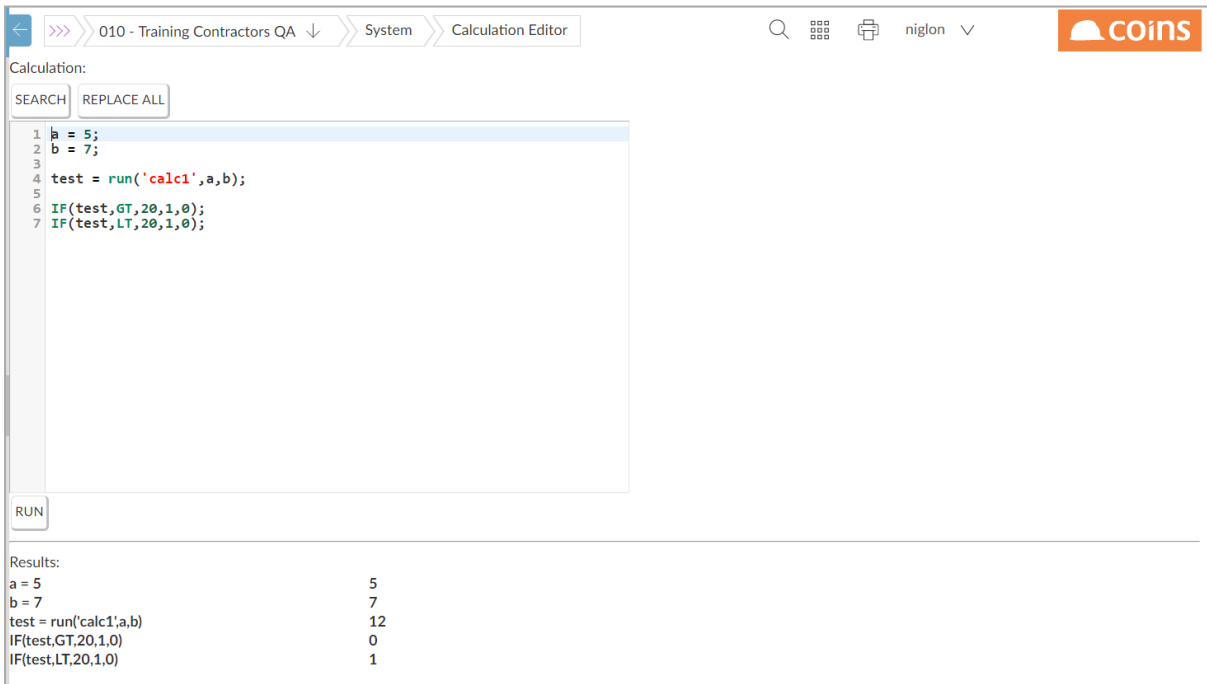
The screenshot shows the COINS Calculation Editor interface. The breadcrumb navigation indicates the path: 010 - Training Contractors QA >> System >> Calculation Editor. The main area contains a code editor with the following script:

```
1 $prog = 'calc1';  
2 a = 5;  
3 b = 7;  
4  
5  
6 run('calc1',a,b);  
7 run('calc1',a,b) + result;  
8 run(prog,a,b);
```

Below the code editor is a 'RUN' button. The results section displays the following output:

\$prog = 'calc1'	calc1
a = 5	5
b = 7	7
run('calc1',a,b)	12
run('calc1',a,b) + result	24
run(prog,a,b)	12

It is also possible to use calculation programs directly within other calculations:



The screenshot shows the COINS Calculation Editor interface. The breadcrumb navigation indicates the path: 010 - Training Contractors QA >> System >> Calculation Editor. The main area contains a code editor with the following script:

```
1 a = 5;  
2 b = 7;  
3  
4 test = run('calc1',a,b);  
5  
6 IF(test,GT,20,1,0);  
7 IF(test,LT,20,1,0);
```

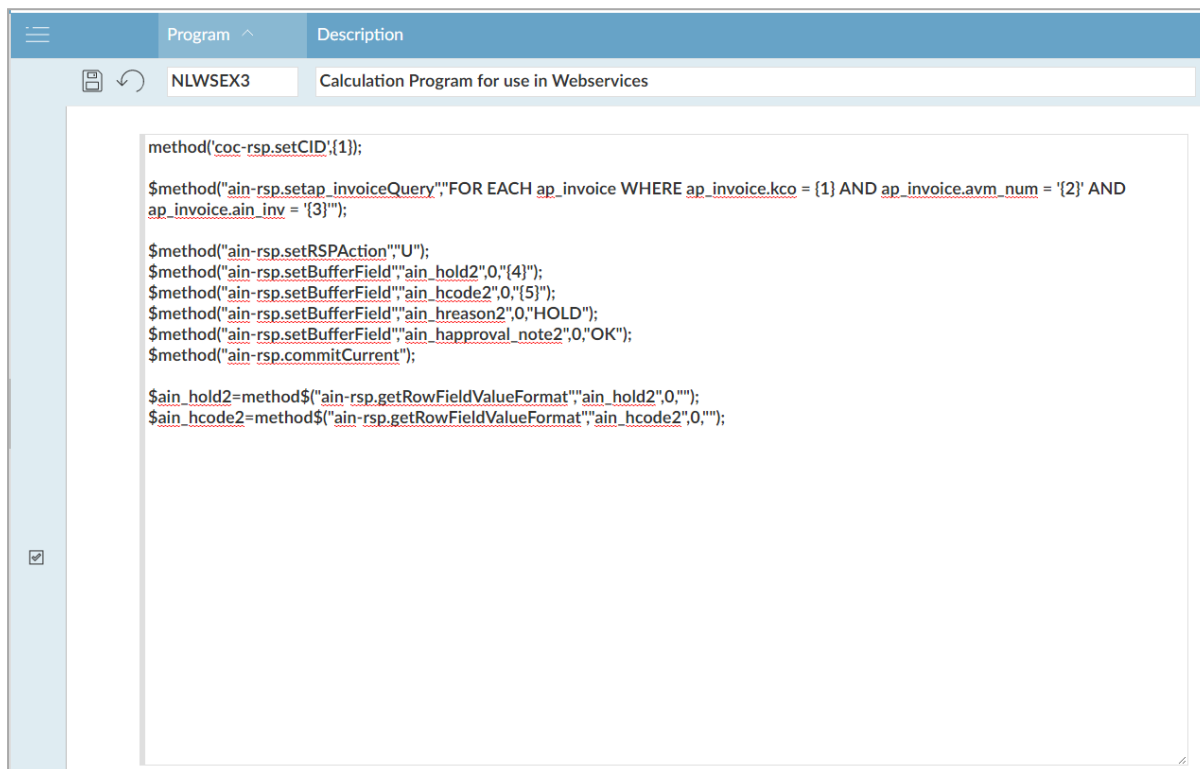
Below the code editor is a 'RUN' button. The results section displays the following output:

a = 5	5
b = 7	7
test = run('calc1',a,b)	12
IF(test,GT,20,1,0)	0
IF(test,LT,20,1,0)	1



### 9.6.3.3 Using Calculation Programs with Web Services

For this example, we are going to setup a calculation program to modify the hold code on an invoice. The program in this example is called NLWSEX3 and has been defined as follows:



The program will accept 5 input variables

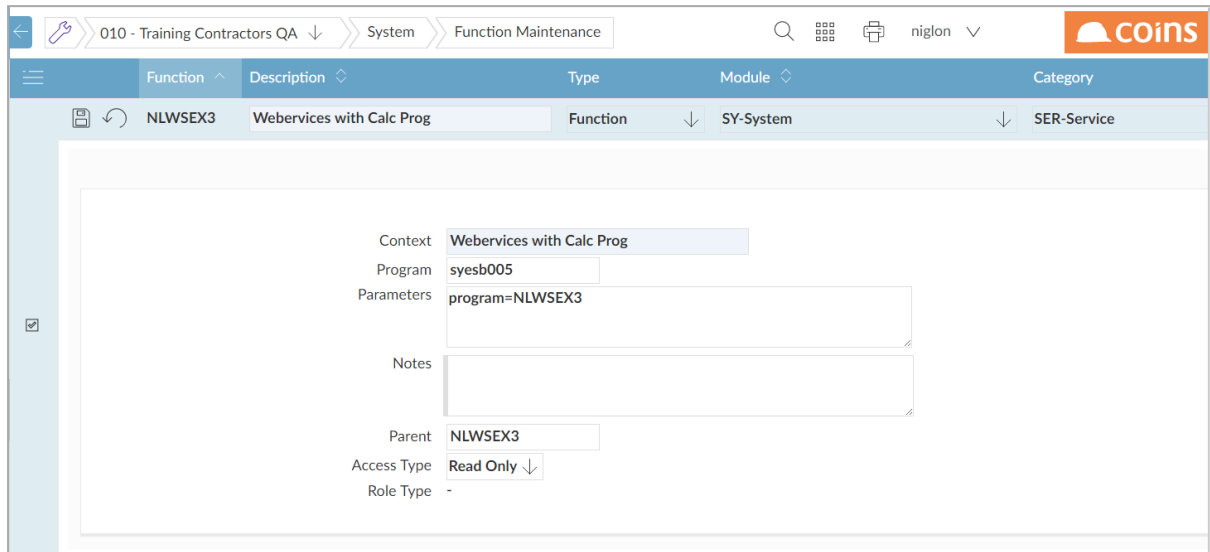
- {1} The COINS Company
- {2} The Supplier Number of the Invoice
- {3} The Invoice Number
- {4} The Hold flag (yes or no)
- {5} The hold code

The calculation will look up the appropriate invoice, set the hold flag and code, and return two variables which will be used in the output message as confirmation of the changes.

#### Create the Function

Create the function called NLWSEX3 with Category of SER – Service and program of syesb005. The Context of the function will determine the description that will appear later on the Web Services menu.

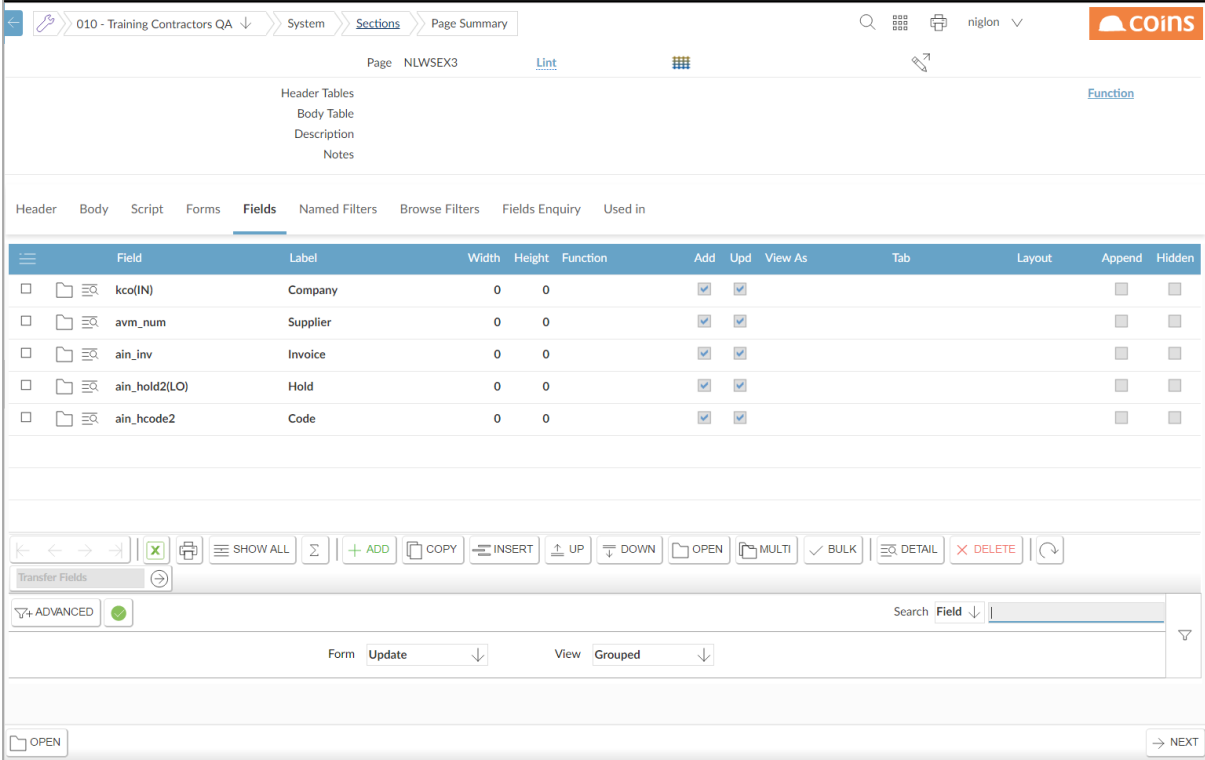
Add the parameter program=NLWSEX3



The program parameter allows you to specify the name of the calculation program to be used. Whilst it is not necessary to call the Calculation Programs the same name as the functions etc. it is probably less confusing if you do keep all the names consistent. The page and the function **MUST** have the same name.

#### Create the Page

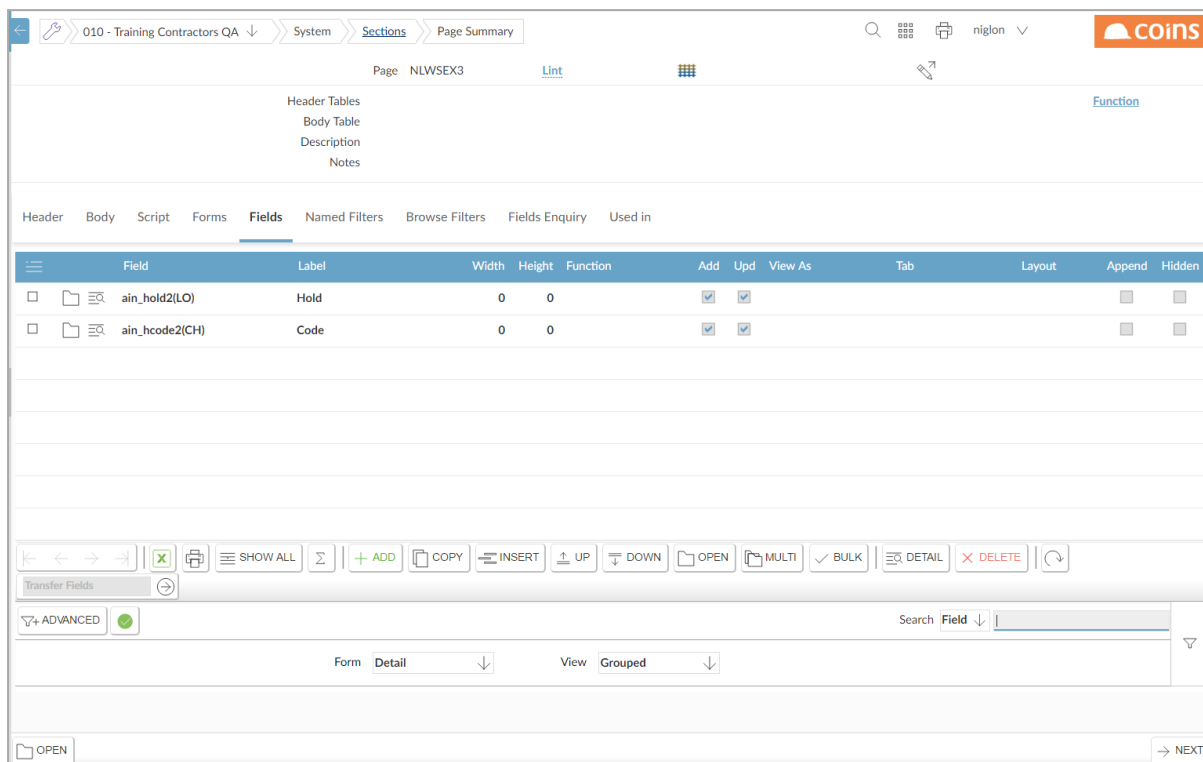
The Page will have two Forms. The Update Form will define the Input Message



Field	Label	Width	Height	Function	Add	Upd	View As	Tab	Layout	Append	Hidden
<input type="checkbox"/> kco(IN)	Company	0	0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> avm_num	Supplier	0	0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> ain_inv	Invoice	0	0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> ain_hold2(LO)	Hold	0	0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> ain_hcode2	Code	0	0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>

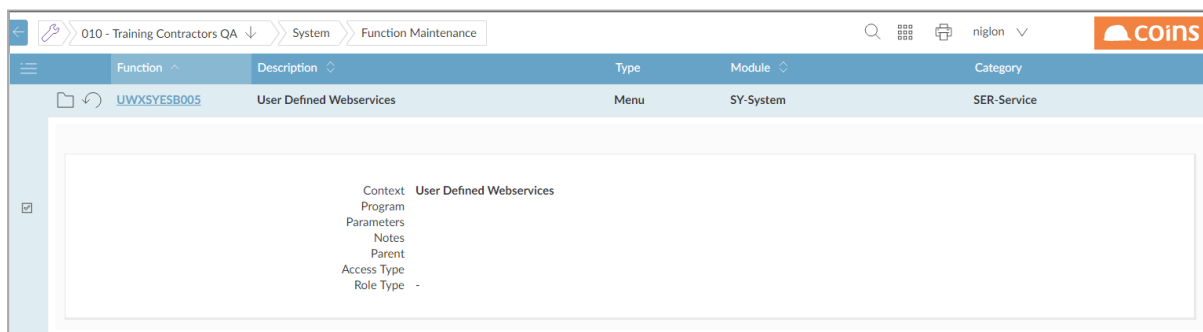
The fields will relate to the input parameters in the calculation program and should be defined in the sequence they are used {1}=kco, {2}=avm\_num etc...

The Detail Form defines the Output message and will return the variables set by the Calculation Program.



### Adding the Function to the Web services Menu

To run the Page Web Service, first check that the menu function UWXSYESB005 exists. If it does not exist, then you need to create it.



Open the Function and select the Menu Items Tab. If the function already exists, there may already be entries on this tab relating to Standard Used Defined Services issued by COINS (prefixed with %) or other User Defined Services created by your company.

Click and enter the name of your new User Defined Services Function (in our example NLWSEX3).



010 - Training Contractors QA > System > Function Maintenance - Function Details

Function UWXSYESB005 User Defined Webservices

Main Menu Items Function Security

Function	Description	Context	Type	Module	Category
<input type="checkbox"/> <a href="#">NLWSEX3</a>	Webervices with Calc Prog		Function	SY-System	Service

SEARCH: Function ↓

Navigate back to the COINS Services Menu and select the SY – System Hyperlink.

## COINS Services

Service	Description
CB	<a href="#">Cash Book</a>
EC	<a href="#">eCommerce</a>
FM	<a href="#">Facilities Management</a>
GL	<a href="#">General Ledger</a>
HS	<a href="#">House Sales</a>
HR	<a href="#">Human Resources</a>
JC	<a href="#">Contract Status</a>
PL	<a href="#">Purchase Ledger</a>
SC	<a href="#">Subcontract Ledger</a>
SL	<a href="#">Sales Ledger</a>
CS	<a href="#">Contract Sales Ledger</a>
SY	<a href="#">System</a>
VF	<a href="#">VAR</a>
PC	<a href="#">Plant</a>
DM	<a href="#">Document Management</a>
PO	<a href="#">Procurement</a>
CI	<a href="#">Company Information Workbench</a>
CC	<a href="#">Customer Care</a>
LA	<a href="#">Land Appraisal</a>
CV	<a href="#">CVR</a>
MK	<a href="#">Marketing</a>
MB	<a href="#">Mobile Applications</a>
SE	<a href="#">SE</a>
ST	<a href="#">Stock</a>

Locate the SYESB00 entries. Your new Function will be in this group.

# COINS Services

[COINS Services](#) >System

Service	Description
SYESB000	<a href="#">Confirmation Message</a>
SYESB001	<a href="#">Login</a>
SYESB002	<a href="#">Get Waiting Message Numbers</a>
SYESB003	<a href="#">Webservices with Dataset</a>
SYESB004	<a href="#">Webservices with Pages</a>
SYESB005	<a href="#">Webservices with Calc Prog</a>
SYESB006	<a href="#">Web service for Appointments/Tasks</a>
SYESB007	<a href="#">Field Get/Set</a>
SYESB011	<a href="#">Save Completed Mobile Form</a>
SYESB012	<a href="#">Database Publish Resend Data</a>
SYESB013	<a href="#">Update User Mobile Configuration</a>
SYESB014	<a href="#">Complete Workflow Stage</a>



Select the Hyperlink to view the service schema.

**COINS Services**

COINS Services > System > Generic Program

Service	SYESB005 (NLWSEX3)
Description	Generic Program
Schema	input.xsd output.xsd exception.xsd acknowledgement.xsd SOAPinput.xsd SOAPoutput.xsd
WSDL	SYESB005.wsdl
WSDLNS	SYESB005.wsdl

The generic program service allows a designer to build an OA calculation program and to call it using this service.  
The argument to this service is the function name of the program to be called.

**Input**

Entity	Type	Documentation
COINSInterface		
Header		
id	string	A unique message identifier from the originating system.
confirm	string	Whether a confirmation message is required. Set to true or yes to have a confirmation message returned. If this is set then an id must also be provided.
action	string	An action type that will indicate why the message was created. For example, this might be CREATE, UPDATE, or DELETE for a message as a result of a maintenance of a record or PUBLISH for the publish of some information.
entity	string	The service that should consume the message.
arguments	string	Arguments that should be passed to the service.
ackID	string	An optional acknowledgement ID. If this ID is set then COINS will respond with an acknowledgement message returning this ackID in the message header.
testMsg	boolean	Whether this message is a test message. A test message will process through in exactly the same way that a normal message would except that at the point where it would normally commit the transaction to the database, the message is then backed out.
UserID	string	The user in the originating system that caused the message to be produced.

The Input Schema - defined by the Update form of the page will look something like this:

**Sample**

```
<COINSInterface>
  <Header id="" confirm="" action="" entity="" arguments="" ackID="" testMsg="true">
    <UserID></UserID>
    <From></From>
    <HostName></HostName>
    <Environment></Environment>
    <Created>2014-04-10T11:54:54.523+01:00</Created>
    <Login>
      <User></User>
      <Password></Password>
      <CID>1</CID>
      <Group></Group>
      <extUser></extUser>
      <extAuth></extAuth>
    </Login>
  </Header>
  <Body>
    <kco>99</kco>
    <avm_num></avm_num>
    <ain_inv></ain_inv>
    <ain_hold2>true</ain_hold2>
    <ain_hcode2></ain_hcode2>
  </Body>
</COINSInterface>
```

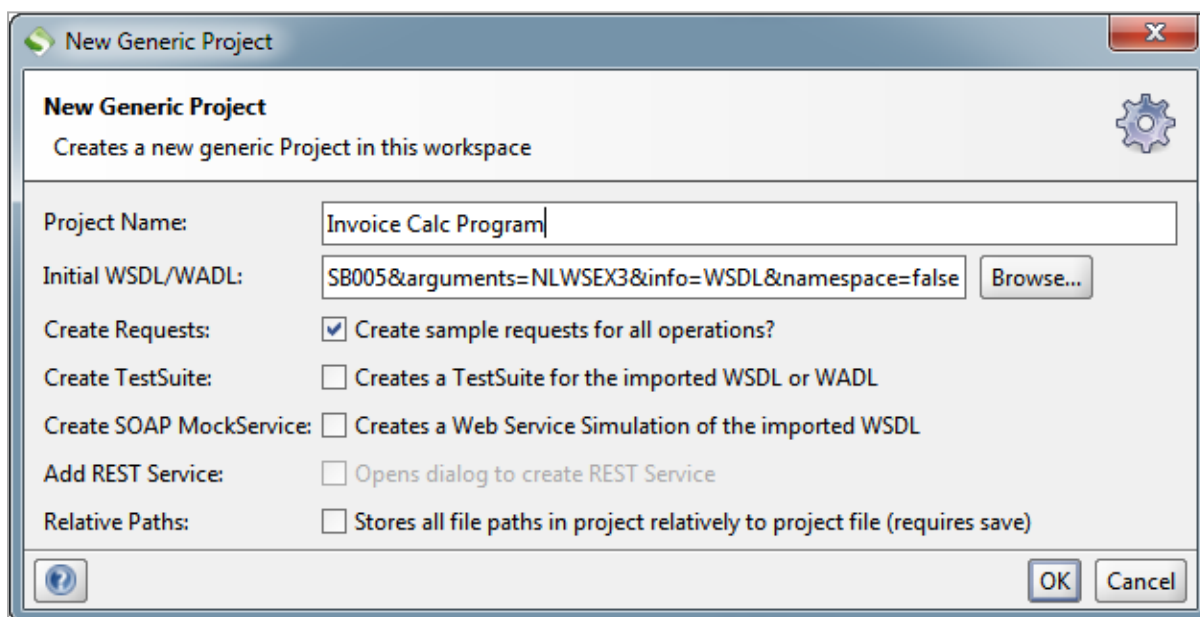
The Output Schema - defined by the Detail Form of the page will look something like this:



### Sample

```
<COINSInterface>
  <Header id="" confirm="" action="" entity="" ackID="">
    <UserID></UserID>
    <From></From>
    <HostName></HostName>
    <Environment></Environment>
    <Created>2014-04-10T11:54:54.605+01:00</Created>
  </Header>
  <Body>
    <ain_hold2>>true</ain_hold2>
    <ain_hcode2></ain_hcode2>
  </Body>
</COINSInterface>
```

#### 9.6.3.4 Testing the Service



Create a new soapUI project and from the Service Schema copy the WSDL shortcut link and paste this into the Initial WSDL field

Click OK

Once the project has been created, run Request 1 in the editor. Replace the Input message with the simplified one from the Input Schema from the Web service.

Enter the hostname, Environment and username/password details.

Delete the lines for Group, extUser and extAuth

Fill in the appropriate Company, Supplier, Invoice and Hold code details for a suitable test invoice. For example:

31-May-2017 Construction Industry Solutions

Page 532 of 600



```
<COINSInterface>
<Header id="" confirm="" action="" entity="" arguments="" ackID="" testMsg="true">
  <UserID></UserID>
  <From></From>
  <HostName>UKSLOUX018.coins.local</HostName>
  <Environment>trainbi</Environment>
  <Created>2014-04-10T09:54:42.130+01:00</Created>
  <Login>
    <User>train1</User>
    <Password>train1</Password>
    <CID>1</CID>
  </Login>
</Header>
<Body>
  <kco>10</kco>
  <avm_num>ABB001</avm_num>
  <ain_inv>11090085</ain_inv>
  <ain_hold2>yes</ain_hold2>
  <ain_hcode2>A2</ain_hcode2>
</Body>
</COINSInterface>
```

Send the message and correct any exceptions if they occur.

If all is correct, a RESPONSE message should be returned.

```
<COINSInterface>
<Header id="" confirm="" action="" entity="" arguments="" ackID="" testMsg="true">
  <UserID></UserID>
  <From></From>
  <HostName>UKSLOUX018.coins.local</HostName>
  <Environment>trainbi</Environment>
  <Created>2014-04-10T09:54:42.130+01:00</Created>
  <Login>
    <User>train1</User>
    <Password>train1</Password>
    <CID>1</CID>
  </Login>
</Header>
<Body>
  <kco>10</kco>
  <avm_num>ABB001</avm_num>
  <ain_inv>11090085</ain_inv>
  <ain_hold2>yes</ain_hold2>
  <ain_hcode2>A2</ain_hcode2>
</Body>
</COINSInterface>

<COINSInterface>
<Header action="RESPONSE" entity="yweb005">
  <UserID>train1</UserID>
  <From>COINS</From>
  <HostName>UKSLOUX018.coins.local</HostName>
  <Environment>trainbi</Environment>
  <Created>2014-04-10T12:14:39.885+01:00</Created>
  </Header>
  <Body>
    <ain_hold2>yes</ain_hold2>
    <ain_hcode2>A2</ain_hcode2>
  </Body>
</COINSInterface>
```

Once you are happy with the response, you can set the testMsg to “false” to apply the change to the database.

Check the invoice in COINS to verify the changes are taking place as expected.

### 9.6.4 Get/Set

Unlike the previous services, Get/Set services do not require functions or pages to be set up as their operation is pre-defined. As suggested by the name, the purpose of a Get/Set service is to retrieve a specific record and set values within it. Get/Set cannot be used to add records, only retrieve/amend existing ones.

From the COINS Services Menu, select the SY – System hyperlink.

# COINS Services

[COINS Services](#) >System

Service	Description
SYESB000	<a href="#">Confirmation Message</a>
SYESB001	<a href="#">Login</a>
SYESB002	<a href="#">Get Waiting Message Numbers</a>
SYESB003	<a href="#">Webservices with Dataset</a>
SYESB004	<a href="#">Webservices with Pages</a>
SYESB006	<a href="#">Web service for Appointments/Tasks</a>
SYESB007	<a href="#">Field Get/Set</a>
SYESB011	<a href="#">Save Completed Mobile Form</a>
SYESB012	<a href="#">Database Publish Resend Data</a>
SYESB013	<a href="#">Update User Mobile Configuration</a>
SYESB014	<a href="#">Complete Workflow Stage</a>



Locate the SYESB007 – Field Get/Set entry

Select the hyperlink to view the service schema.

**COINS Services**

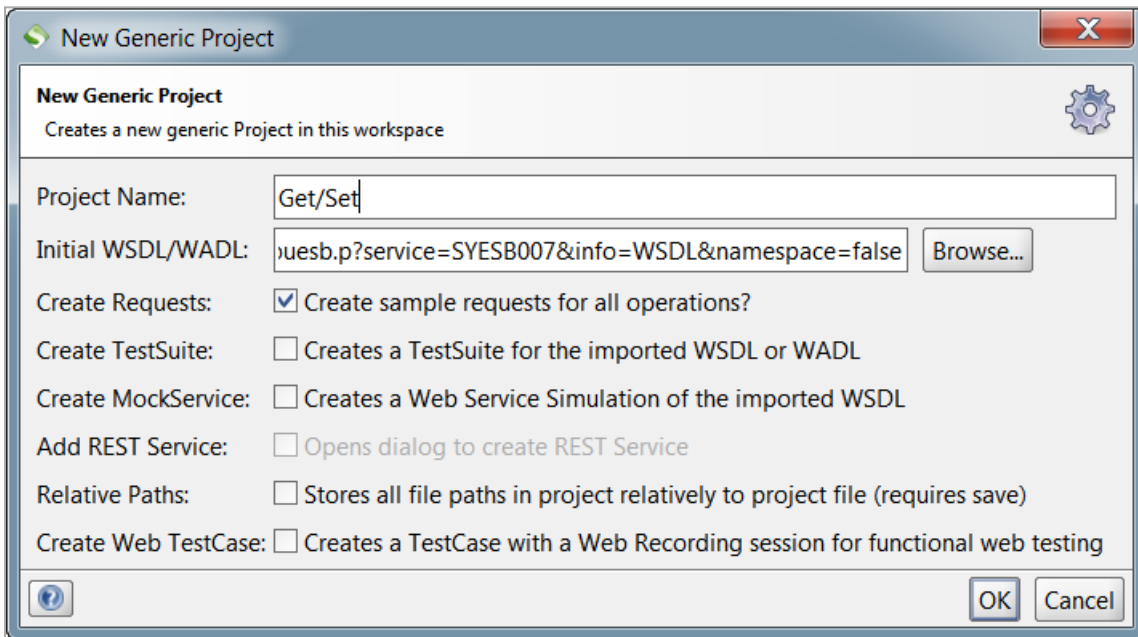
[COINS Services](#) > [System](#) > [Field Get/Set](#)

<b>Service</b>	SYESB007
<b>Description</b>	Field Get/Set
<b>Schema</b>	<a href="#">input.xsd</a> <a href="#">output.xsd</a> <a href="#">exception.xsd</a> <a href="#">acknowledgement.xsd</a> <a href="#">SOAPInput.xsd</a> <a href="#">SOAPOutput.xsd</a>
<b>WSDL</b>	<a href="#">SYESB007.wsdl</a>
<b>WSDL NS</b>	<a href="#">SYESB007.wsdl</a>

The generic table field service allows a service client to request fields from and/or update fields in a specific record.  
The criteria specified to find the record must result in a single record being returned.  
If set field name and value pairs are specified then they are set before the get returns the current values.

**Input**

Entity	Type	Documentation
COINSInterface		
Header		
id	string	A unique message identifier from the originating system.
confirm	string	Whether a confirmation message is required. Set to true or yes to have a confirmation message returned. If this is set then an id must also be provided.
action	string	An action type that will indicate why the message was created. For example, this might be CREATE, UPDATE, or DELETE for a message as a result of a maintenance of a record or PUBLISH for the publish of some information.
entity	string	The service that should consume the message.
arguments	string	Arguments that should be passed to the service.
ackID	string	An optional acknowledgement ID. If this ID is set then COINS will respond with an acknowledgement message returning this ackID in the message header.
testMsg	boolean	Whether this message is a test message. A test message will process through in exactly the same way that a normal message would except that at the point where it would normally commit the transaction to the database, the message is then backed out.
UserID	string	The user in the originating system that caused the message to be produced.
From	string	The name of the system that produced the message.



**New Generic Project**

Creates a new generic Project in this workspace

Project Name:

Initial WSDL/WADL:

Create Requests:  Create sample requests for all operations?

Create TestSuite:  Creates a TestSuite for the imported WSDL or WADL

Create MockService:  Creates a Web Service Simulation of the imported WSDL

Add REST Service:  Opens dialog to create REST Service

Relative Paths:  Stores all file paths in project relatively to project file (requires save)

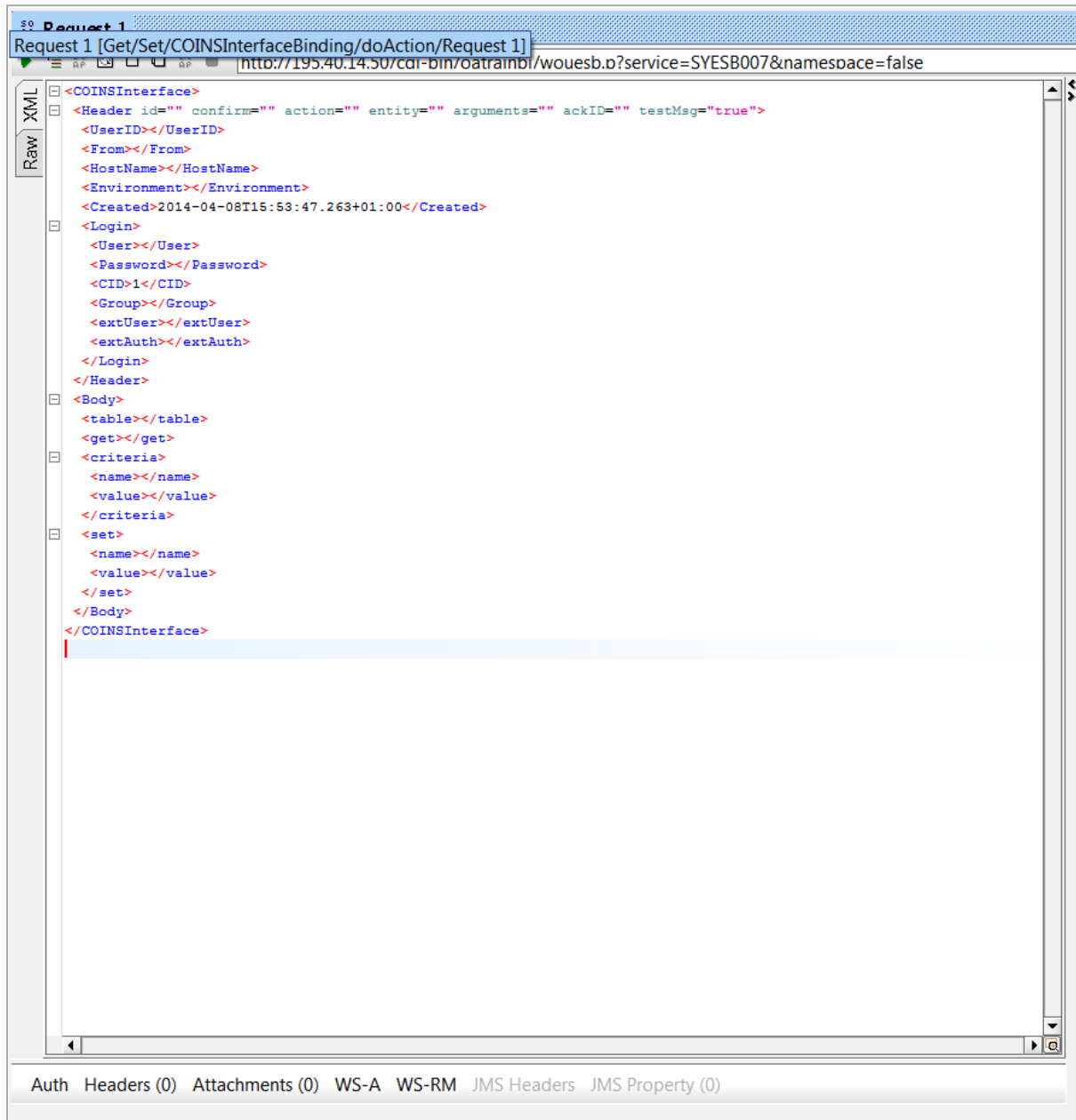
Create Web TestCase:  Creates a TestCase with a Web Recording session for functional web testing

Create a new soapUI project and, from the service schema, copy the WSDL shortcut link into the initial WSDL field.

Click OK

Once the project has been created, run Request1 in the editor.

Replace the Input message in the editor with the simplified version from the Input Message Schema



Enter the hostname, Environment and username/password detail.

Delete the lines for Group, extUser and extAuth

In the Body section of the Input Message, the first entries are:

```
<table></table>
<get></get>
```

These allow you to specify the table the record will come from and the fields that will be returned in the Output Message. The get statement is a comma separated list of fields.

The next lines relate to the criteria which will identify the record to be retrieved. The entries require the name of the key field and the value to be used to identify the record.

```
<criteria>  
  <name></name>  
  <value></value>  
</criteria>
```

If there is more than one key field which identifies a record, you will need to copy the Criteria block for each required field. For example, for a table with two key fields:

```
<criteria>  
  <name></name>  
  <value></value>  
</criteria>  
<criteria>  
  <name></name>  
  <value></value>  
</criteria>
```

Finally, the Set block of lines allow you to specify the field to be changed and the value to which it should be set.

```
<set>  
  <name></name>  
  <value></value>  
</set>
```

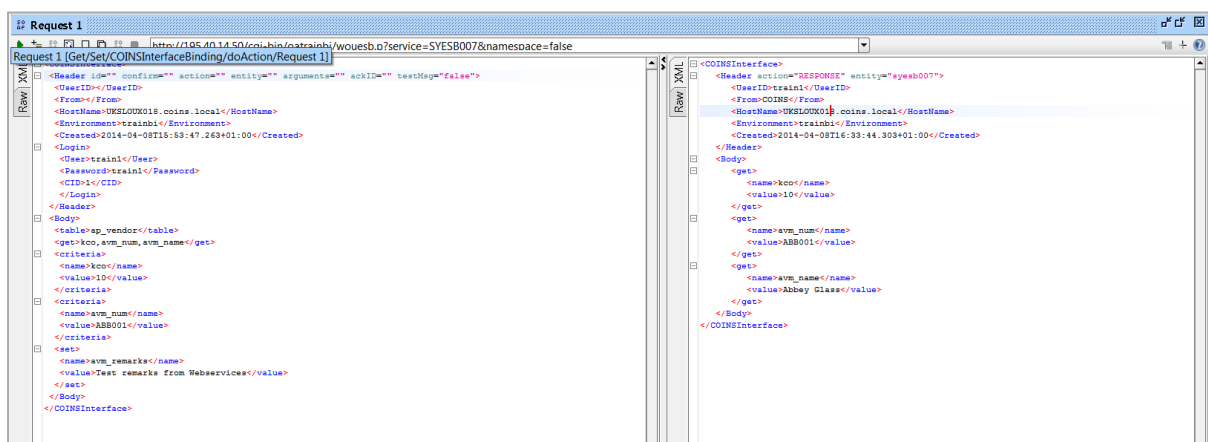
As with the Criteria block, if more than one field is to be changed, there should be a block of set lines for each field.

In the following example, a supplier record in the ap\_vendor table will be retrieved and the remarks field in the supplier record will be updated.



```
<COINSInterface>
<Header id="" confirm="" action="" entity="" arguments="" ackID="" testMsg="false">
  <UserID></UserID>
  <From></From>
  <HostName>UKSLOUX018.coins.local</HostName>
  <Environment>trainbi</Environment>
  <Created>2014-04-08T15:53:47.263+01:00</Created>
  <Login>
    <User>train1</User>
    <Password>train1</Password>
    <CID>1</CID>
  </Login>
</Header>
<Body>
  <table>ap_vendor</table>
  <get>kco, avm_num, avm_name</get>
  <criteria>
    <name>kco</name>
    <value>10</value>
  </criteria>
  <criteria>
    <name>avm_num</name>
    <value>ABB001</value>
  </criteria>
  <set>
    <name>avm_remarks</name>
    <value>Test remarks from Webservices</value>
  </set>
</Body>
</COINSInterface>
```

Sending this message should produce the following return with the Output message showing the three fields we requested in the get entry of the Input Message.



Remember that leaving the Test Message entry set to “true” will process the messages but will not apply the changes to the database. The test message must be set to “false” for database changes to occur.





If the database change is successful our example will have the following effect on the supplier record:

Main Terms Configuration Payment Method Analysis Sets Insurance Notes

Supplier Account: ABB001

Payee Name: Abbey Glass  
Name: Abbey Glass

Address: 42 Bramall Lane  
Sheffield

Postcode: S25 4DL (SHEFFIELD)

Search Name: Abbey Glass  
Short Name: Abbey

Telephone: 01642 887766

Default Print Method:  
Fax: 01642 887767  
Email: tim.drake@coins-global.com  
Contact:

Supplier's Code for Us:

Country: GB-United Kingdom  
VAT Registration No: 762070208 - VAT - D1 - Purchase Standard Rate @47.5%

Remarks: Test remarks from Webservices

Account Currency: GBP-Sterling Allow Other Currencies:

Account Closed:  Account Dormant:  Factor:

## 10 Building COINS Desktops - Overview

This guide is intended for anyone with OA Designer skills to create and maintain COINS Desktops.

This guide is not intended for end-users for which Desktop Wizards are available to build desktops from pre-defined components. End User documentation is available from the OA Navigation section of the COINS Learning Resources on the COINS Client Area:

[www.coins-global.com](http://www.coins-global.com)

The guide will work through the creation of an example Desktop. It covers the creation of the Desktop structure and explores the configuration of the various Tile types that are available.

## 10.1 Desktop Structure

To configure a user's desktop, set up a new three-level menu function.

- The first level is the desktop itself. This is the function specified on the user record(
- The second level must contain one or more menus which correspond to the tabs on the desktop.
- Each of these "tab" menus must contain one or more menu functions with a new type: Section. These correspond to the collapsible sections on the desktop. If a tab has only one section, the section is displayed without a title.
- The menu items on the section menu correspond to the tiles. Desktop Menu Item Maintenance allows you to configure which function the tile runs, how the function opens, and what the tile looks like.

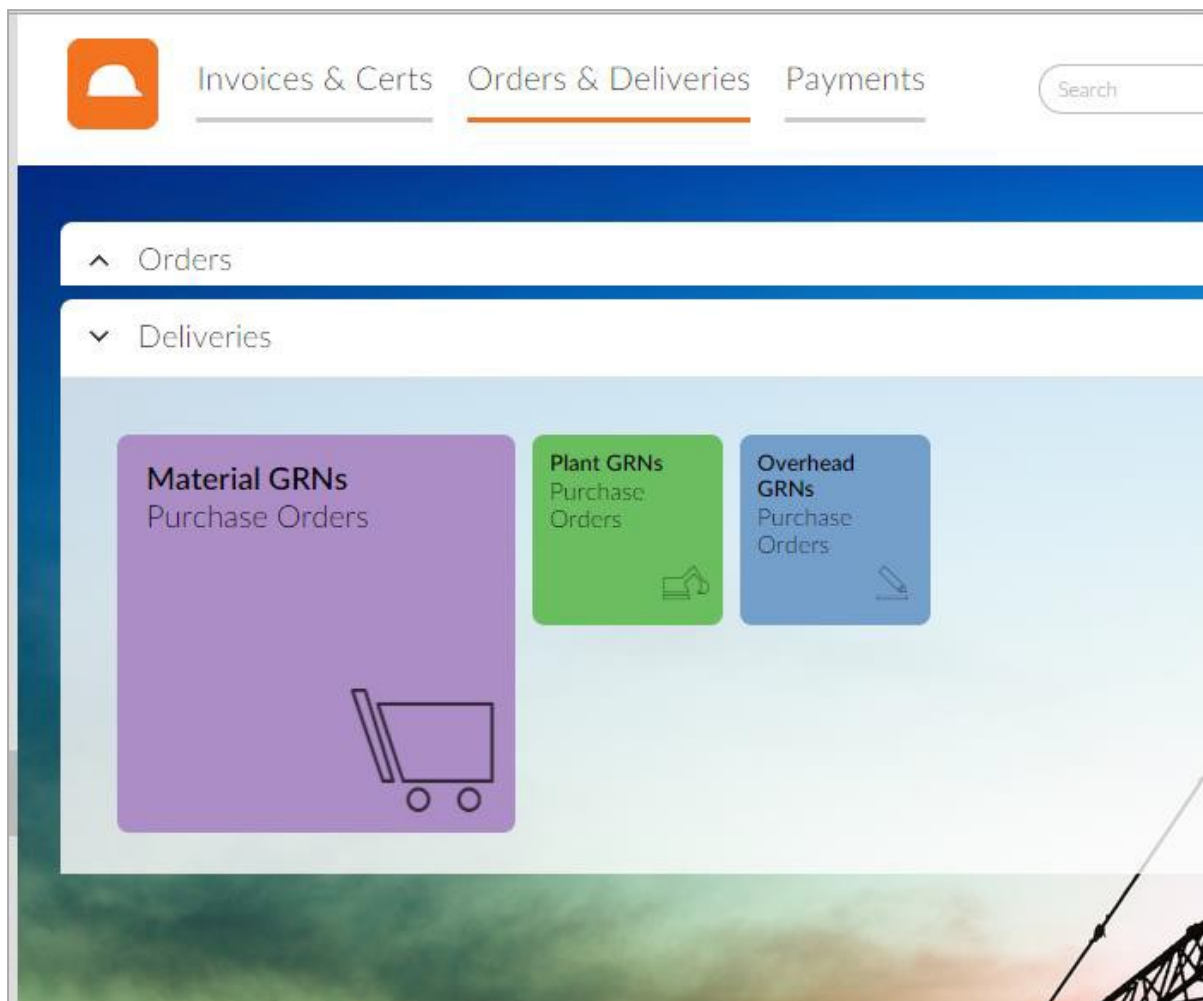
If the user configures their own desktop, COINS automatically creates functions with the following naming convention:

Tabs - the function code of the desktop menu + T + n (where n is the number of the tab).

Sections - the function code of the tab + S + n (where n is the number of the section on the tab)

.

Desktop Menu for User JSMITH



Function	Explanation
JSMITH	Desktop Menu
JSMITHT1	First Tab
JSMITHT2	Second Tab
JSMITHT2S1	First Section on Second Tab
JSMITHT2S1	Second Section on Second Tab
%WPOxxxxxxx	Functions on Section
%WPLxxxxxxx	
%WSCxxxxxxx	
%WJCxxxxxxx	

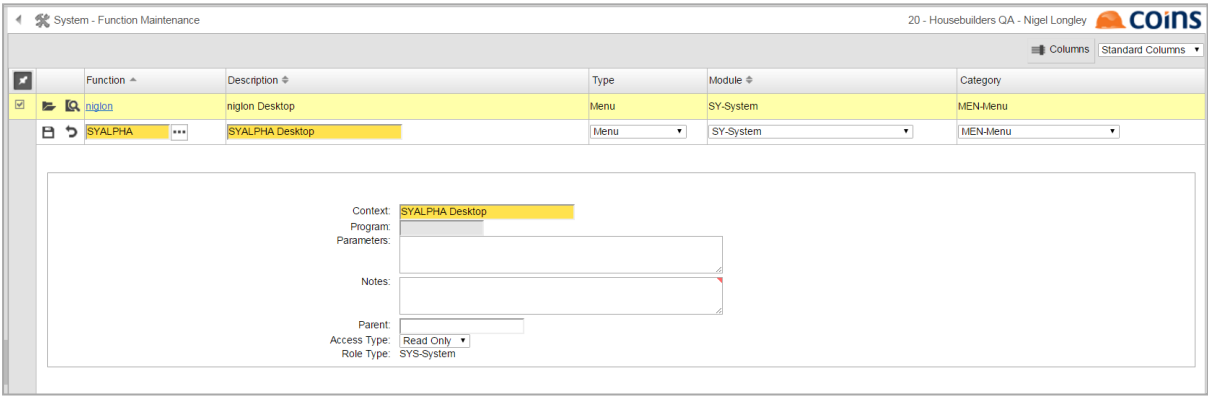
## 10.1.1 Creating the Functions (Manually)

A desktop wizard is available to automate much of this work, however to ensure a good understanding of how the desktop works, we will build it manually so that we can explore each of the components.

For our example, we are going to create a Desktop called SYALPHA with two Tabs and two sections on each tab

### 10.1.1.1 Desktop Functions

First we need to create the Top Level Desktop Function.



Function	Description	Type	Module	Category
niglon Desktop	niglon Desktop	Menu	SY-System	MEN-Menu
SYALPHA Desktop	SYALPHA Desktop	Menu	SY-System	MEN-Menu

Context: SYALPHA Desktop  
Program:  
Parameters:  
Notes:  
Parent:  
Access Type: Read Only  
Role Type: SYS-System

Ensure that the module is SY-System and the Category is MEN-Menu.

Click Save

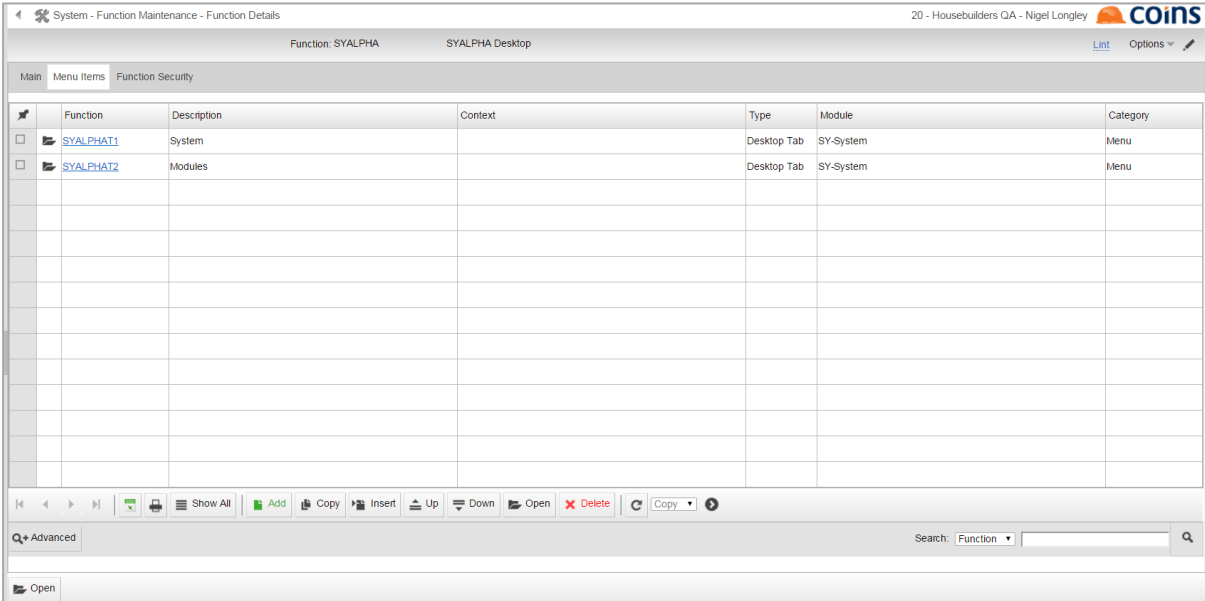
### 10.1.1.1 Desktop Tab Functions

- Next we can create the two Tabs. We will have two Tabs, one called System and the other called Modules
- Copy the Top Level Function and add T1 to the function name for the first tab, and add T2 to the name of the second function for the second tab.
- Change the Type to Desktop Tab
- Change the Description and Context fields to System and Modules respectively.

Function	Description	Type	Module	Category
SYALPHA	SYALPHA Desktop	Menu	SY-System	MEN+Menu
SYALPHAT1	System	Desktop Tab	SY-System	MEN+Menu
<p>Context: System                      Program:                      Parameters:                      Notes:                      Parent:                      Access Type: Read Only                      Role Type: SYS-System</p>				
SYALPHAT2	Modules	Desktop Tab	SY-System	MEN+Menu
<p>Context: Modules                      Program:                      Parameters:                      Notes:                      Parent:                      Access Type: Read Only                      Role Type: SYS-System</p>				

Now we need to add our tabs to the top level.

- Click on the hyper link for function SYALPHA and click on the menu items tab.
- Add the functions SYALPHAT1 and SYALPHAT2 to the list.



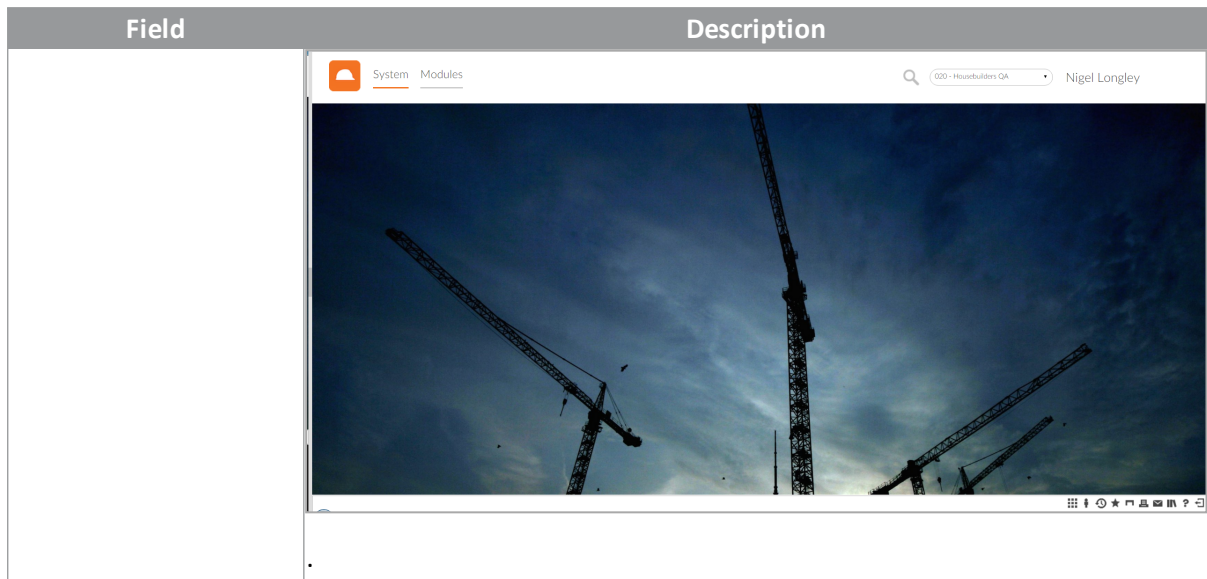
Function	Description	Context	Type	Module	Category
SYALPHAT1	System		Desktop Tab	SY-System	Menu
SYALPHAT2	Modules		Desktop Tab	SY-System	Menu

We can now test the initial design

### 10.1.2 Add the Desktop to a user profile

The desktop needs to be configured for each user that will use it. Different users can use the same desktop, or each user can have their own desktop. Configuring the desktop involves setting up a menu and specifying the menu on the user record.





If a Desktop has been set up for a user, but they have a user home page set, when they log in they will see the home page (but will still be able to access the Desktop). If you clear the home page from their user record, they will see the Desktop when they log in.



## 10.1.3 Desktop Section Functions

Currently we only have the desktop and two tabs. Now we can look at creating the sections.

In Function Maintenance, create two new functions SYALPHAT1S1 and SYALPHAT1S2

For both, set the Type to Desktop Section

For SYALPHAT1S1 the description and Context should be Utilities

For SYALPHAT1S2 the description and Context should be Maintenance

Function	Description	Type	Module	Category
SYALPHA	SYALPHA Desktop	Menu	SY-System	MEN-Menu
SYALPHAT1	System	Desktop Tab	SY-System	MEN-Menu
SYALPHAT1S1	Utilities	Desktop Section	SY-System	MEN-Menu
SYALPHAT1S2	Maintenance	Desktop Section	SY-System	MEN-Menu

<p>Context: Maintenance                  Program:                  Parameters:                  Notes:                  Parent:                  Access Type: Read Only                  Role Type: SYS-System</p>
--

SYALPHAT2	Modules	Desktop Tab	SY-System	MEN-Menu
-----------	---------	-------------	-----------	----------

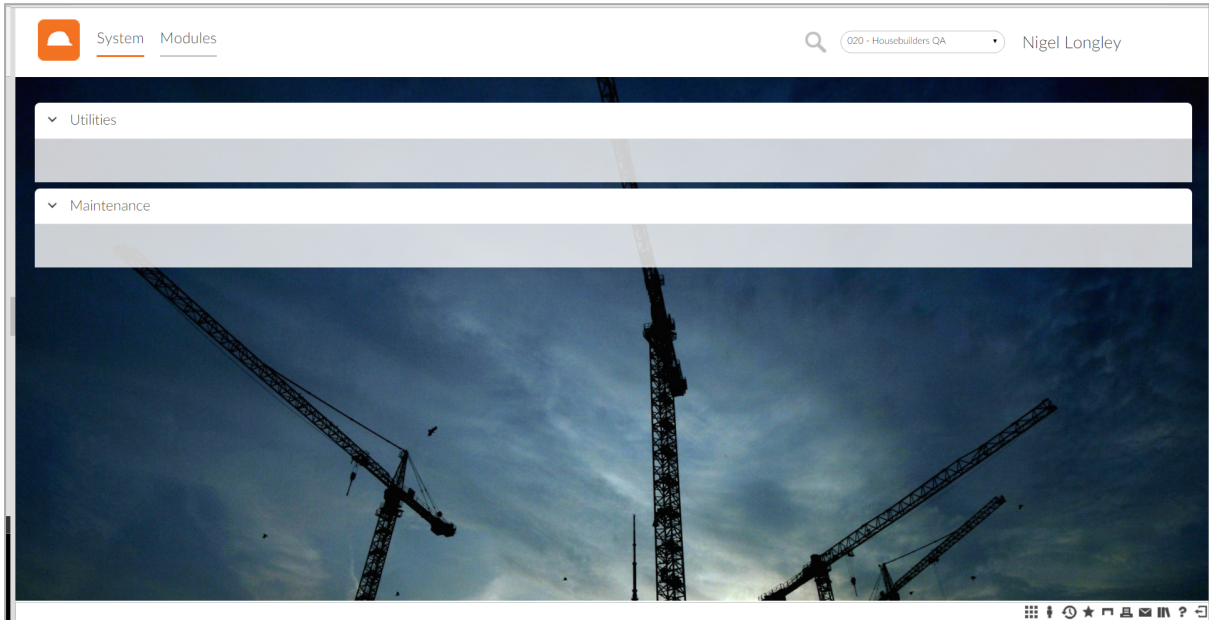
  

<p>Context: Modules                  Program:                  Parameters:                  Notes:                  Parent:                  Access Type: Read Only                  Role Type: SYS-System</p>
--

Add these two functions to the Menu Items Tab of SYALPHAT1

Function	Description	Context	Type	Module	Category
SYALPHAT1S1	Utilities		Desktop Section	SY-System	Menu
SYALPHAT1S2	Maintenance		Desktop Section	SY-System	Menu

Test the design for your test user.

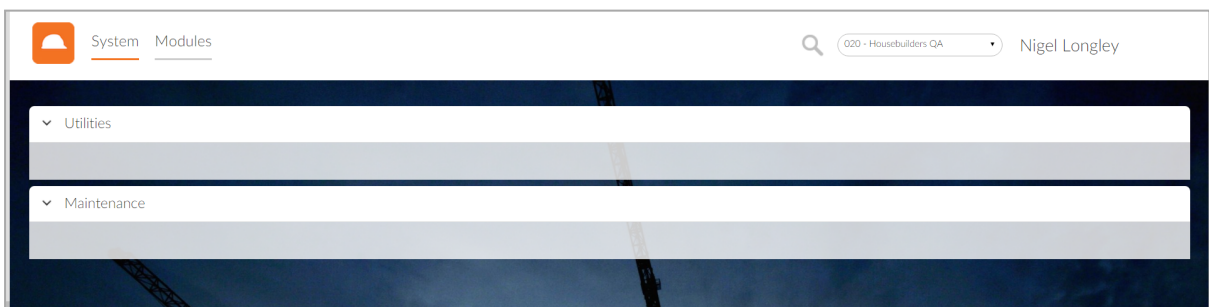


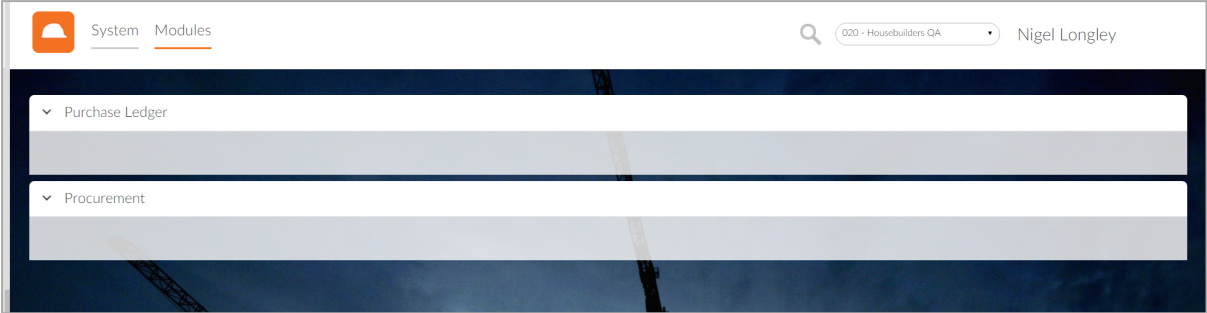
Repeat the steps above to create sections for the Second Tab (SYALPHAT2S1 and SYALPHAS2T2)

For SYALPHAT2S1 the description and Context should be Purchase Ledger

For SYALPHAT2S2 the description and Context should be Procurement

Test your design



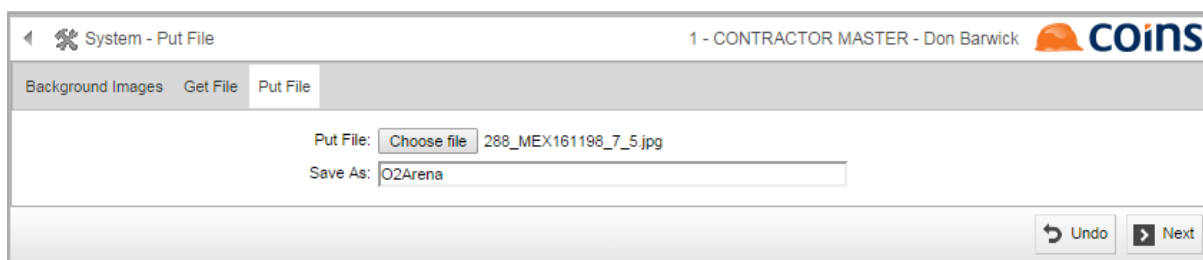


## 10.1.4 Uploading Desktop Images

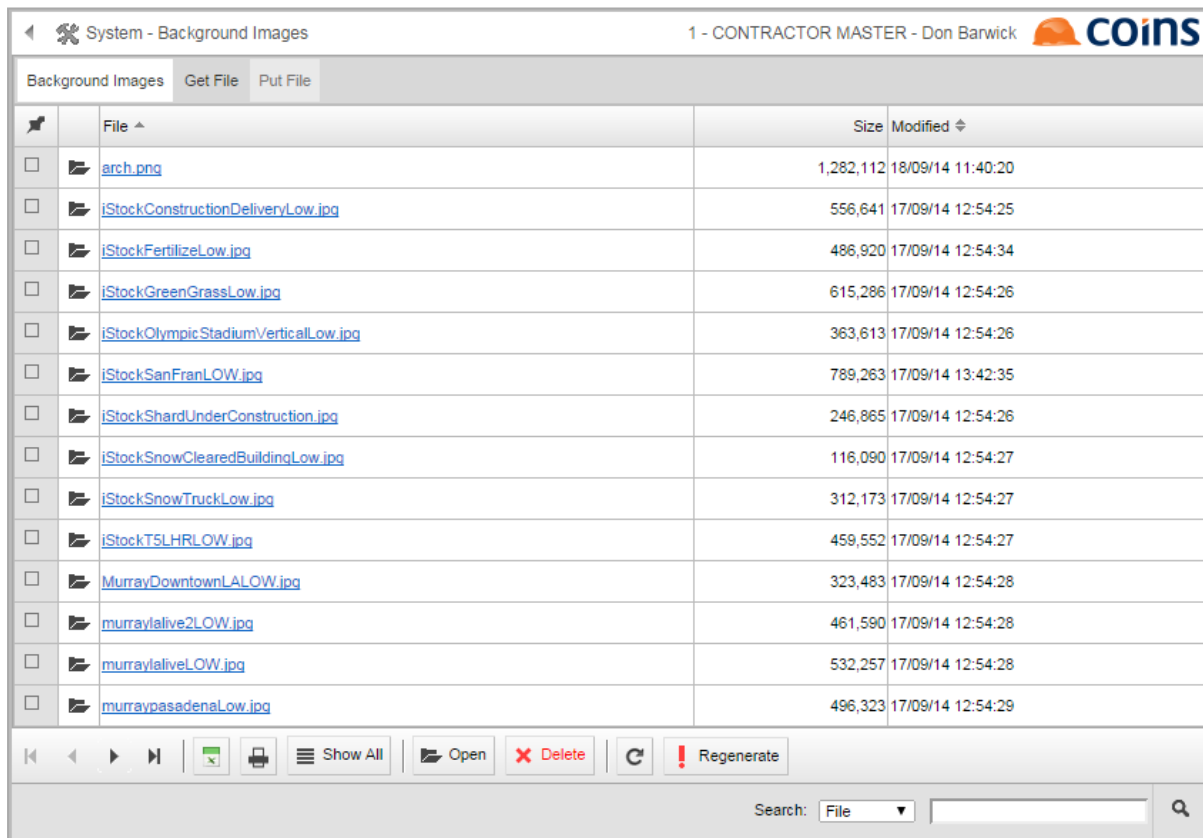
Two new functions allow the system administrator to upload images used on the desktop.

- System > System Setup > Background Images allows you to load images that can be used as the desktop background. Individual users can then select one of these to use on their desktop.

### Background Images – Put File Tab



The file name you give to the file is used in the Background Image selection list in User Maintenance. The images are saved in the \$BASE/custimage/bg/ directory. Some standard images are provided in the \$BASE/images/bg/ directory; you can copy these or upload them using Background Images.



- System > System Setup > User Images allows you to load the images that are used as the account picture on each user’s desktop. The image files must be JPG format files, and should be approximately 60px square. (You can use larger images but the browser will resize them, which does not always produce good results.) You must name the files

<userid>.jpg (for example, for a user whose user ID is JSMITH, the file would be jsmith.jpg. The images are saved in the \$BASE/custimage/sysuser/ directory.



If there is no file for a user, the account image will be the image from the user’s COINS HR record if one exists, otherwise it will be a silhouette image.

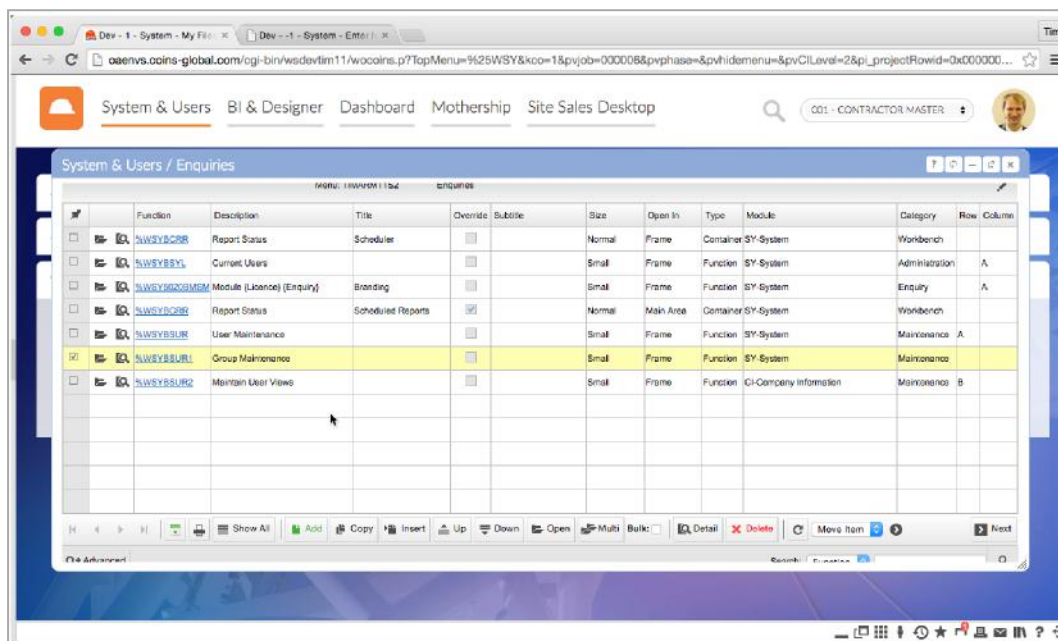
Account Image Silhouette

Zak Barlow



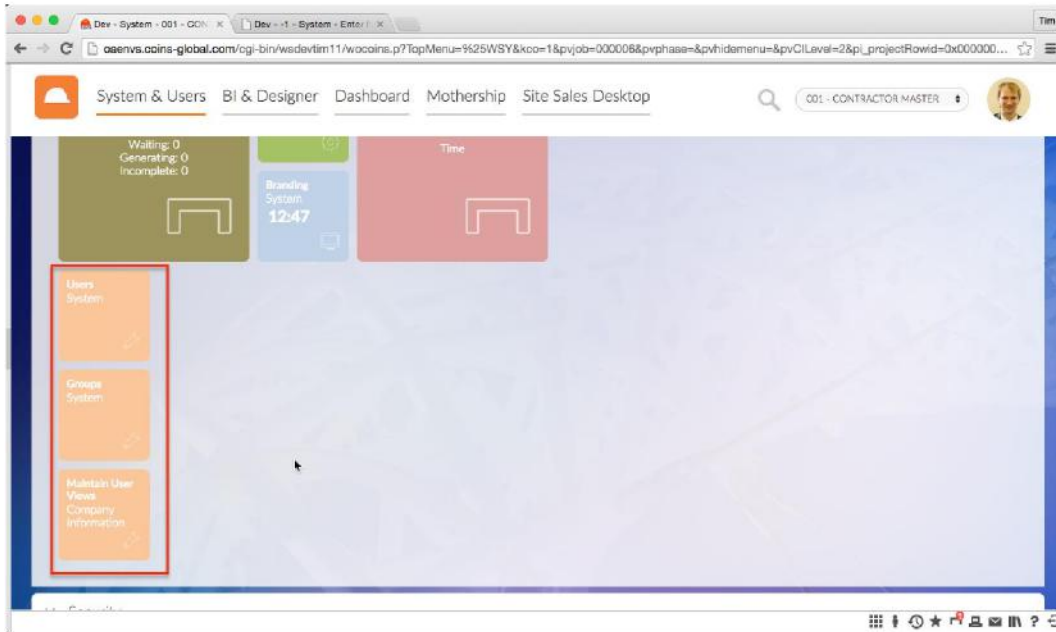
## 10.1.5 Tile Rows and Columns

On tile maintenance it is now possible to group tiles into rows or columns, by assigning codes to tiles.



The Row code allows row breaks to be inserted. If a row code changes from one item to the next then a new row is started. (This can already be achieved by inserting a blank function to cause a new row to start but this is introduced to provide equivalent of the column code; see below.)

### 10.1.5. Desktop – Tile Rows

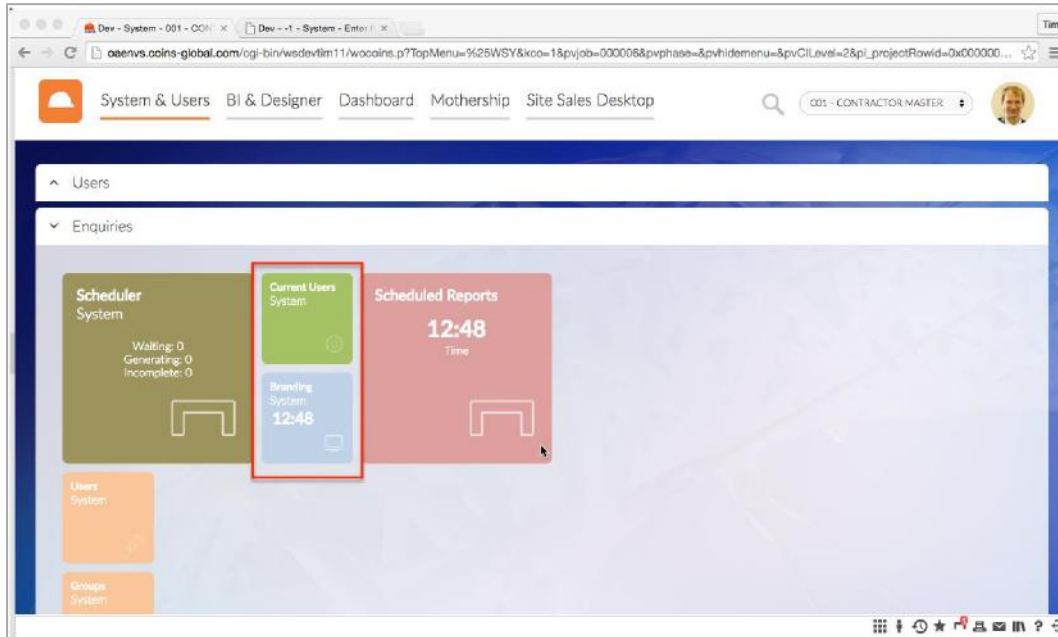


In the example above a new row will start for User Maintenance (A), Group Maintenance (Blank) and Maintain User Views (B).

It is also possible to group tiles in to columns.

Where a set of tiles shares a common Column code then they are set out in a single column.

### 10.1.5. Desktop – Tile Columns



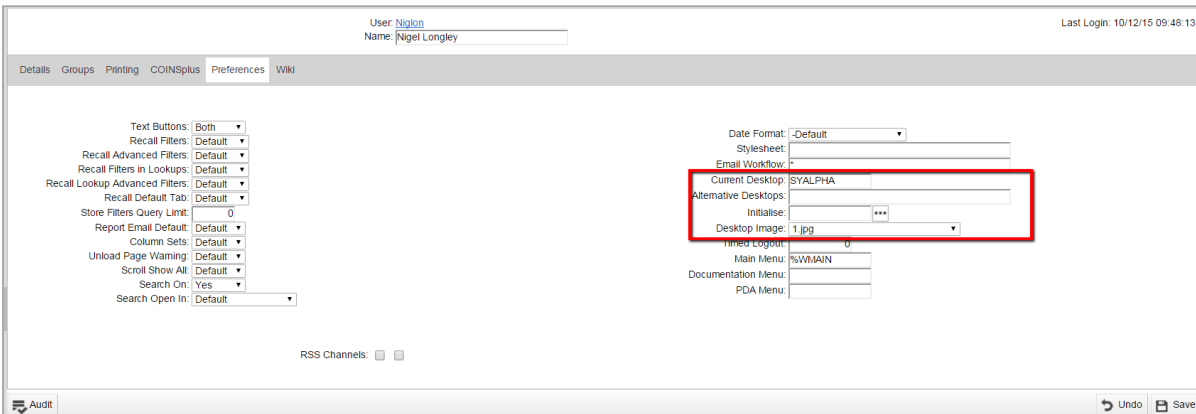
In the example above, Current Users and Module Licence Enquiry share the same column code, “A”, so are grouped in a single column on the desktop. Columns are only grouped for non-blank column codes.



## 10.1.6 Add the Desktop to a user profile

The desktop needs to be configured for each user that will use it. Different users can use the same desktop, or each user can have their own desktop. Configuring the desktop involves setting up a menu and specifying the menu on the user record.

In System > User Maintenance > Users select a user (or create a new account for test purposes) and select the User Preferences tab.



The screenshot shows the 'User Preferences' tab for user 'Nigel Longley'. The 'Current Desktop' field is highlighted with a red box and contains the value 'SYALPHA'. Other fields include 'Date Format', 'Stylesheet', 'Email Workflow', 'Alternative Desktops', 'Initialise', 'Desktop Image', 'Timed Logout', 'Main Menu', 'Documentation Menu', and 'PDA Menu'. The 'Desktop Image' field is set to '1.jpg'.

File	Description
Desktop	<p>Allows the system administrator to specify the menu function that displays the desktop for this user. If the menu function does not already exist, COINS creates a new function; this in effect creates an empty Desktop, which either the system administrator or the user can configure.</p> <p style="text-align: center;">For our example, enter SYALPHA as the desktop name.</p> <p>If this is left blank, the user has no Desktop set up, and the Set Homepage button is available to allow the user to select a single function as their homepage.</p>
Desktop Image	<p>Allows the system administrator to specify the background image for this user's desktop. This is a drop-down list of the available image files. Standard images are available, and administrators can upload additional ones (see later).</p> <p>Choose an existing image for your desktop.</p> <p>Login as your test user. The desktop should be displayed on login</p>



File	Description
------	-------------



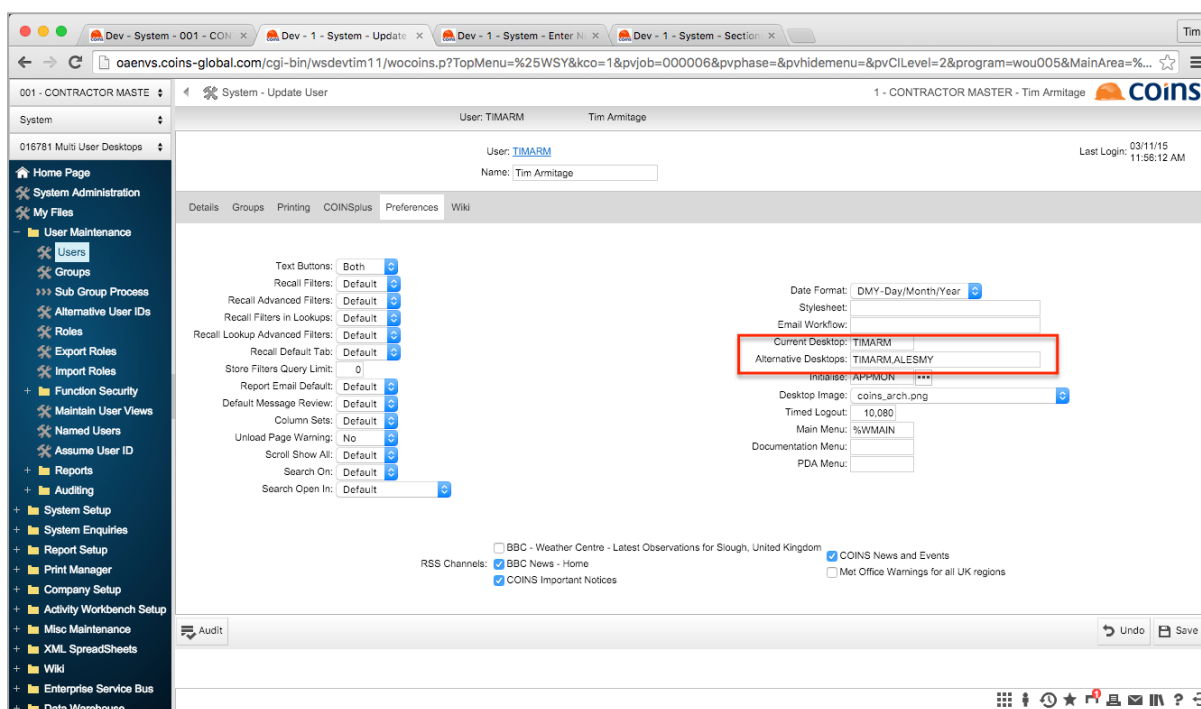
If a Desktop has been set up for a user, but they have a user home page set, when they log in they will see the home page (but will still be able to access the Desktop). If you clear the home page from their user record, they will see the Desktop when they log in.

## 10.1.7 Alternative Desktops

You can setup more than one desktop (which you can think of as roles of the user or personas of the user) the user then can select from the available desktops switching their role/persona. The tabs, sections and items on the desktop refresh based on the latest selected role/persona.

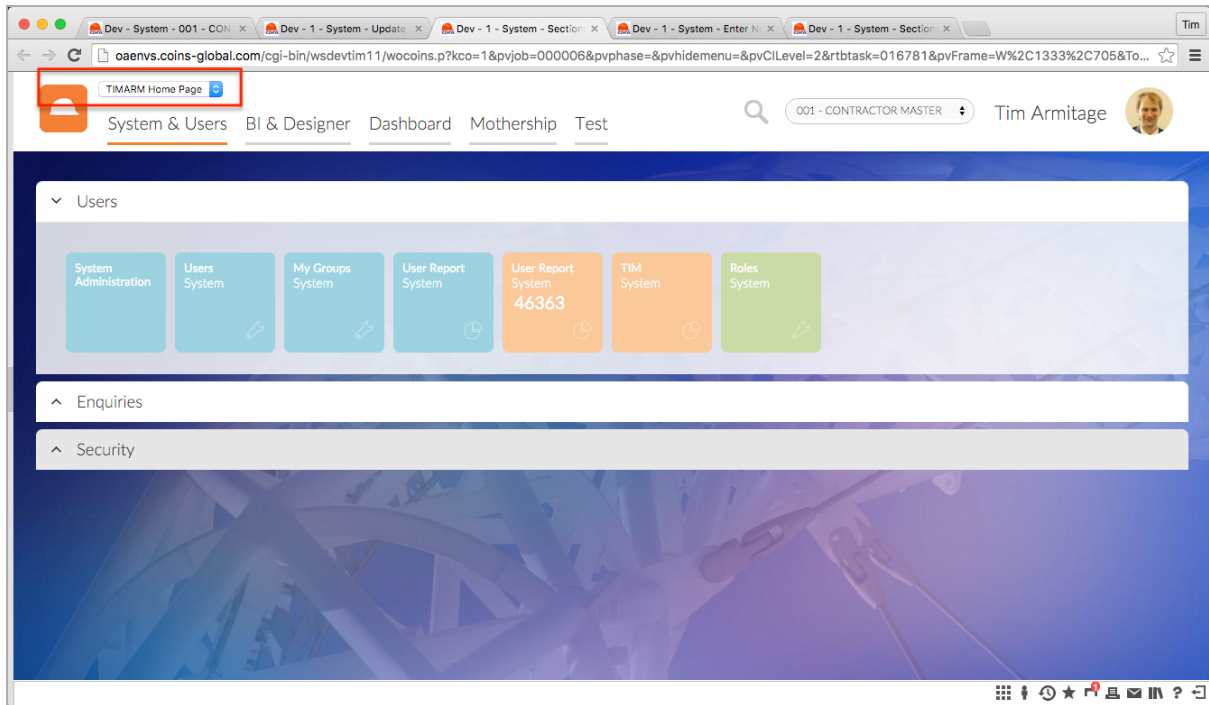
### 10.1.7.1 User Maintenance

A new option is introduced on the user maintenance page

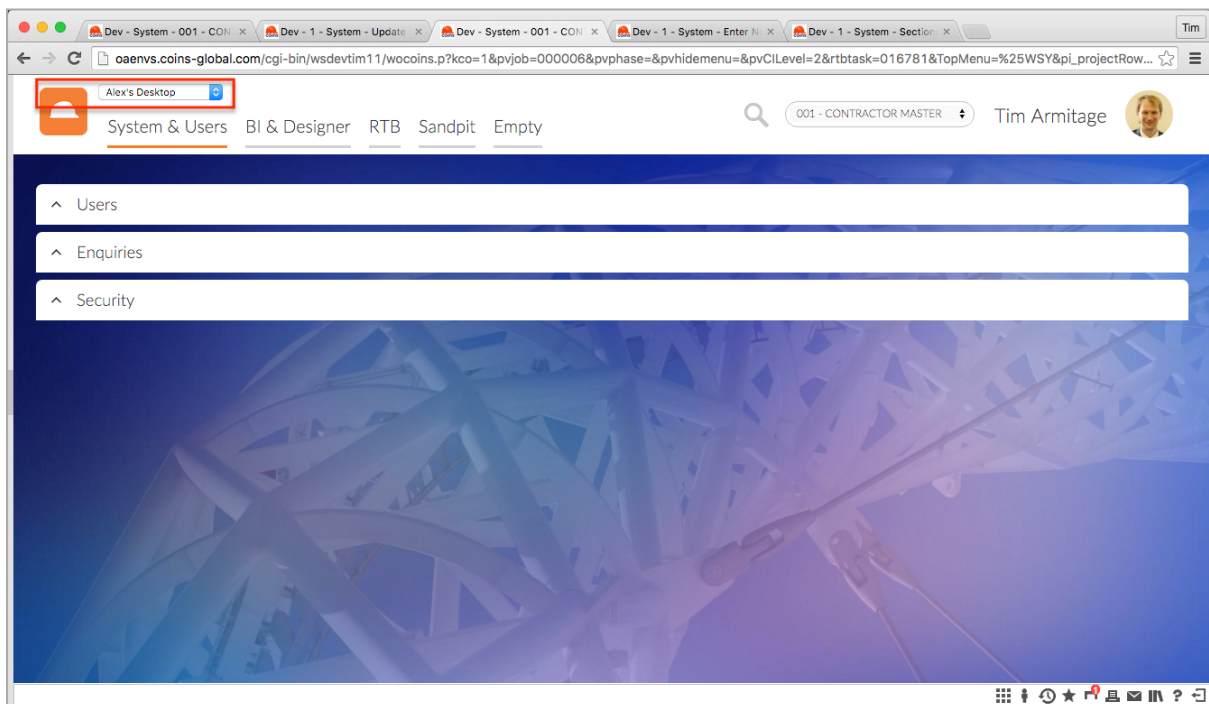


The alternative desktops is a list of desktops that will already exist that the user can switch to. The currently selected desktop is shown in the existing desktop field. This is the one that the user will see if they login (or refresh) the COINS page.

The alternative desktops are shown at the top left of the desktop above the tabs.



As soon as you select an alternative (examples are similar but different tabs, sections and tiles) your user record is changed so that this becomes your default desktop and the page is refreshed showing you the new desktop.

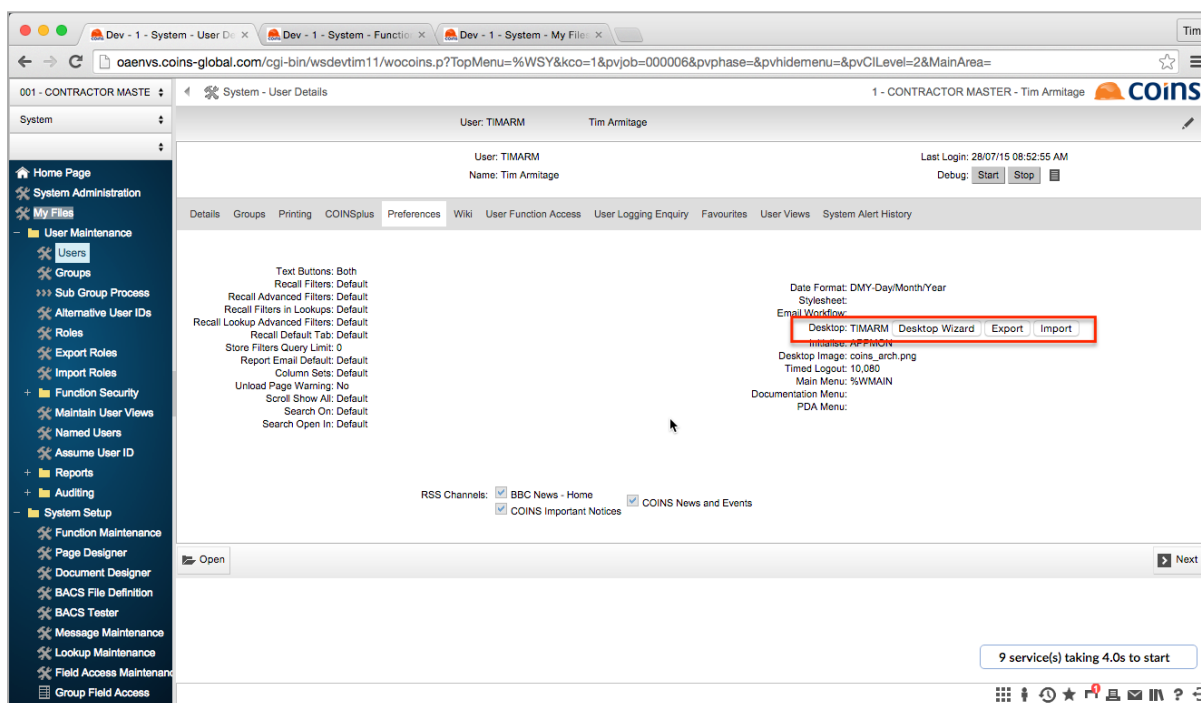


## 10.1.8 Desktop Import/Export

Desktops can be maintained in COINS and are retained as functions, menus and menu items. It is possible to copy the functions, menus and menu items using standard function export and import features. From 11.04 new functionality makes the export and import of desktops between users and/or between environments easier.

### 10.1.8.1 User Maintenance

Two new buttons have been added to user maintenance. A desktop export and a desktop import button alongside the Desktop field on the summary page.



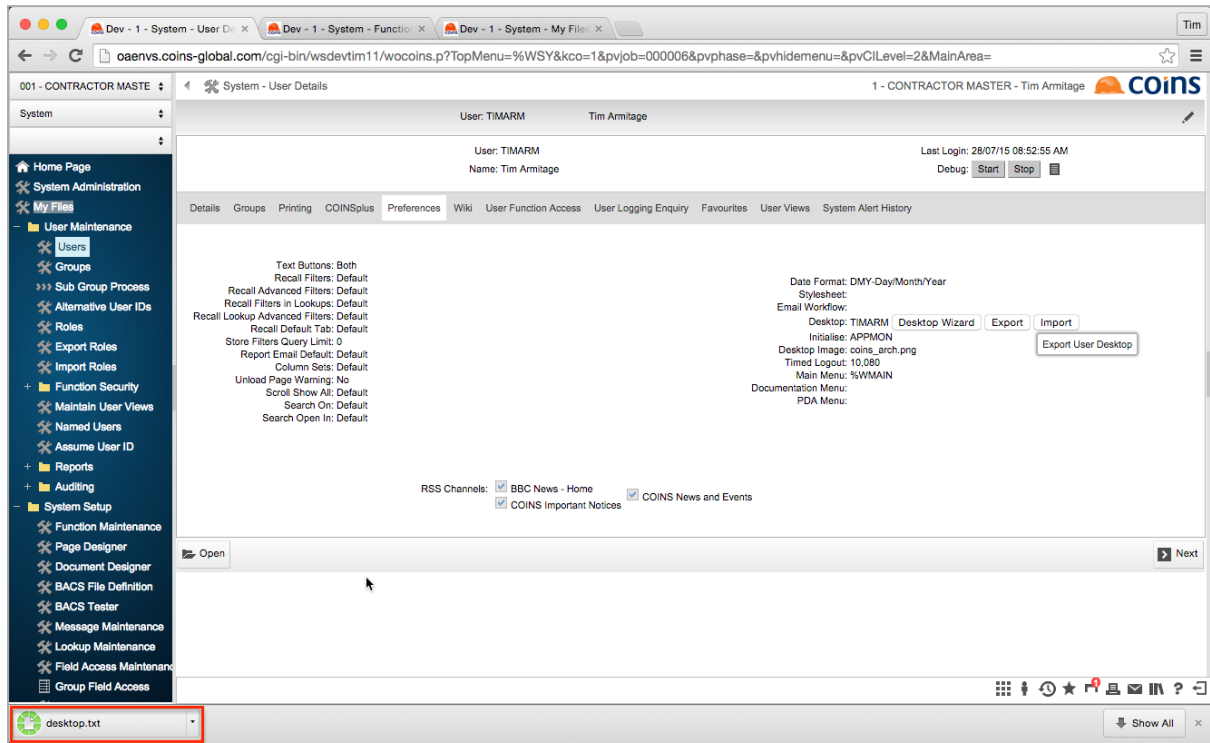
These can be used to export the user's desktop data (menus and menu items) to a file and then to upload this data either to another user or to another environment.

### 10.1.8.2 Export

Pressing the export button causes a download of the menu and menu item data for the current user.

This finds the desktop menu (in this case TIMARM) and exports it together with all the submenus (sections and tabs) and the items on those menus. Only menus (and contents) of user specific menus are exported i.e. beginning TIMARM. This is to prevent any standard menus being copied (and overwritten). It is expected that standard menus (tabs/sections)

would be copied using existing standard techniques. The menus containing the reference to the standard menus is exported.



The export file contains the original desktop code (TIMARM) and an export of the functions, menus and items that make it up.

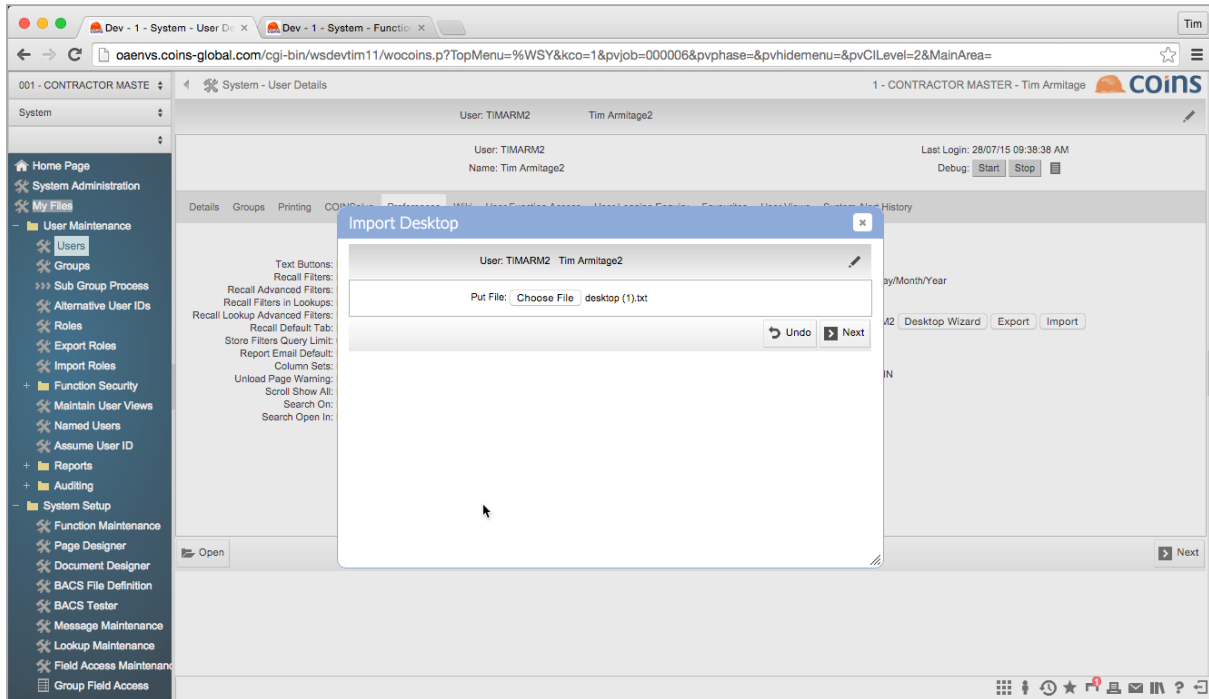


```
desktop.txt
"TIMARM"
"TIMARM" "TIMARM Home Page" "" "" "" 0 "" "SY" "Menu" "TIMARM" "TIMARM" "MEN" "TIMARM Home
Page" yes "" "SYS" 20140709.63729 "TIMARM" "11.02" no "" "" ""
"2014-07-09T17:42:090x0000000001d31884" "" "" ""
"TIMARMT1" "System & Users" "" "" "" 0 "R" "SY" "Menu" "TIMARMT1" "" "MEN" "System &
Users" yes "" "SYS" 20141218.30591 "TIMARM" "11.03" no "" "" ""
"2014-12-18T08:29:510x000000000215ea05" "" "" ""
"TIMARMT1S1" "Users" "" "" "" 0 "R" "SY" "Section" "TIMARMT1S1" "" "MEN" "Users" yes ""
"SYS" 20141218.30591 "TIMARM" "11.03" no "" "" "" "2014-12-18T08:29:510x000000000215ea08"
"" "" ""
"TIMARMT1S2" "Enquiries" "" "" "" 0 "R" "SY" "Section" "TIMARMT1S2" "" "MEN" "Enquiries"
yes "" "SYS" 20141218.30591 "TIMARM" "11.03" no "" "" ""
"2014-12-18T08:29:510x000000000215eb10" "" "" ""
"TIMARMT1S3" "Security" "" "" "" 0 "R" "SY" "Section" "TIMARMT1S3" "" "MEN" "Security" yes
"" "SYS" 20141218.30591 "TIMARM" "11.03" no "" "" ""
"2014-12-18T08:29:510x000000000215eb06" "" "" ""
"TIMARMT2" "BI & Designer" "" "" "" 0 "R" "SY" "Menu" "TIMARMT2" "" "MEN" "BI & Designer"
yes "" "SYS" 20141218.30591 "TIMARM" "11.03" no "" "" ""
"2014-12-18T08:29:510x000000000215eb0a" "" "" ""
"TIMARMT2S1" "Maintenance" "" "" "" 0 "R" "SY" "Section" "TIMARMT2S1" "" "MEN"
"Maintenance" yes "" "SYS" 20141218.30591 "TIMARM" "11.03" no "" "" ""
"2014-12-18T08:29:510x000000000215eb0d" "" "" ""
"TIMARMT2S2" "Utilties" "" "" "" 0 "R" "SY" "Section" "TIMARMT2S2" "" "MEN" "Utilties" yes
"" "SYS" 20141218.30591 "TIMARM" "11.03" no "" "" ""
"2014-12-18T08:29:510x000000000215eb15" "" "" ""
"TIMARMT3" "Dashboard" "" "" "" 0 "R" "SY" "Menu" "TIMARMT3" "" "MEN" "Dashboard" yes ""
"SYS" 20141218.30591 "TIMARM" "11.03" no "" "" "" "2014-12-18T08:29:510x000000000215eb81"
"" "" ""
"TIMARMT3S1" "Activity" "" "" "" 0 "R" "SY" "Section" "TIMARMT3S1" "" "MEN" "Activity" yes
"" "SYS" 20141218.30591 "TIMARM" "11.03" no "" "" ""
```



## 10.1.8.3 Import

Pressing the import button brings up an upload dialog.



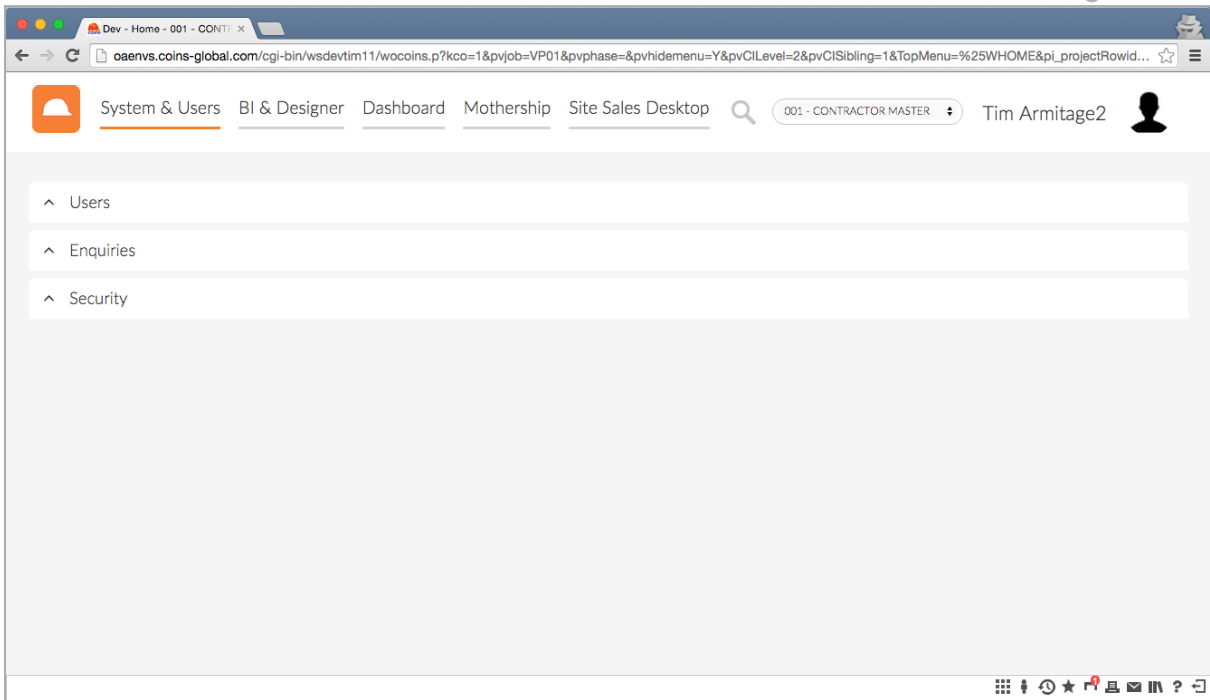
Selecting the export file previous downloaded will cause the desktop that was exported to be imported as the desktop for the selected user. All the menus and functions that are extensions of the desktop menu will be renamed for the new desktop menu.

e.g. TIMARMT1 would be renamed TIMARM2T1

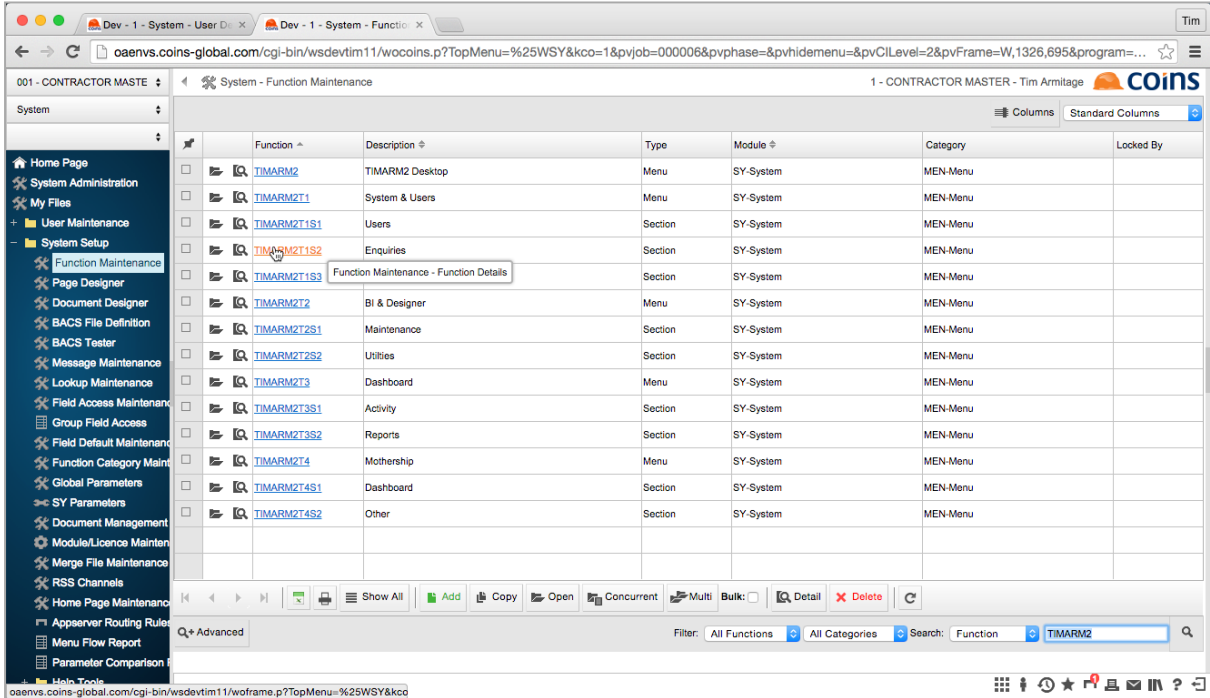
TIMARMT1S1 would be renamed TIMARM2T1S1

This applies to the menus and the items (sub menus) on them.

Having imported the TIMARM desktop the TIMARM2 desktop looks the same



This is copy of the original and TIMARM2\* functions, tabs and sections now exist.



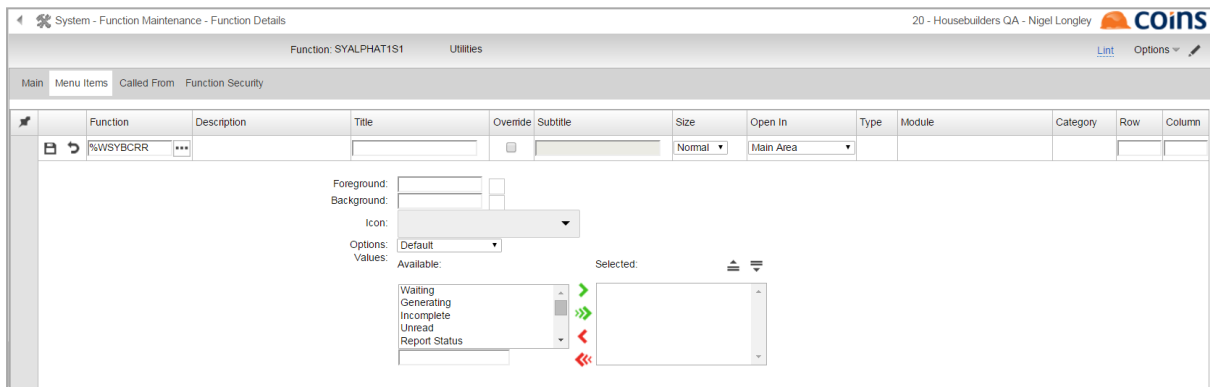


### 10.1 Smart Tiles

Now that the basic structure of the desktop is in place we can start to add tiles to the sections. First of all we will add a smart tile to a standard coins function.

Smart Tiles show summary information about a function (such as financial values or the number of items requiring attention). The values on these tiles are kept up to date and refreshed automatically.

On the Desktop Section Function SYALPHAST1, open the Menu Items Tab and add the function %WSYBCRR (Report Status Workbench).

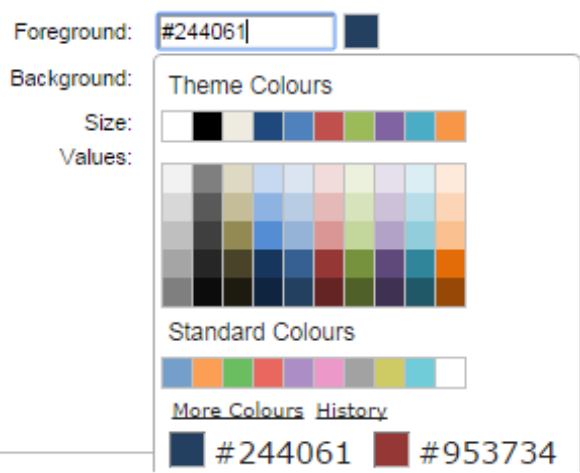


This additional fields allow you to configure the tile on the section.

For each tile, you can specify the following:

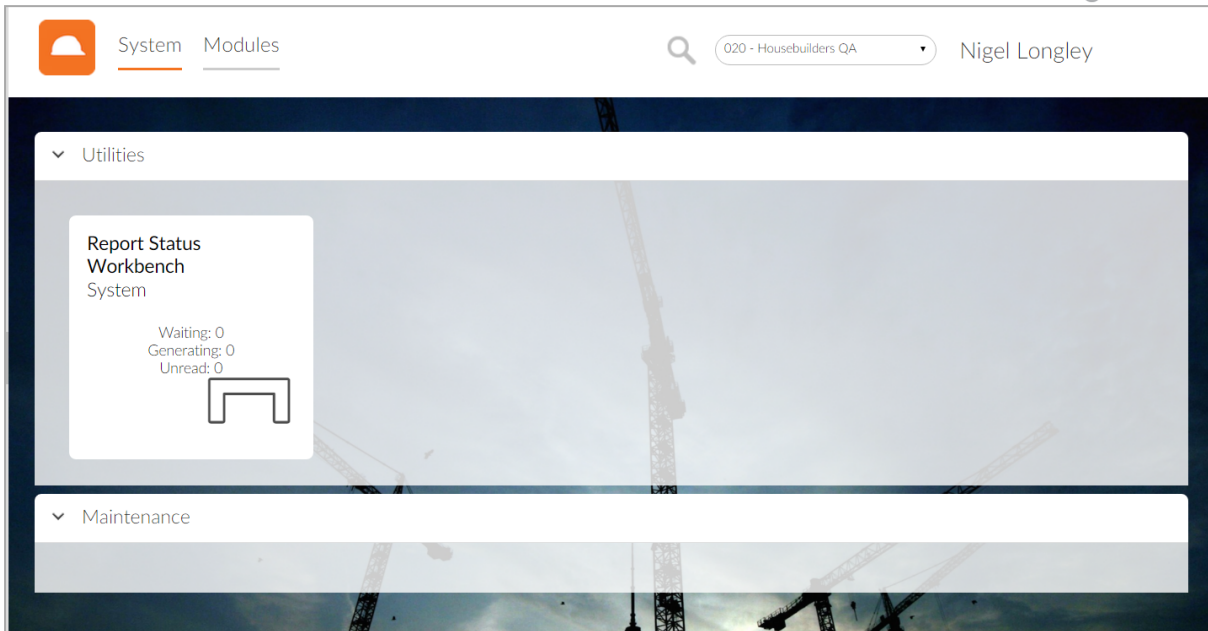
Field	Description
Function	The COINS function that the tile runs.  You can leave this blank (that is, enter a line without a function) to force the following tiles to display on a new line.
Title	The description to display on the tile. If blank, the tile shows the standard context description.
Override	Whether to override the subtitle of the tile. If this is not ticked, the module the function belongs to is used as the subtitle.
Subtitle	The subtitle to display on the tile. If blank (and if Override is ticked), the subtitle is blank.
Size	The size of the tile (small, wide or normal). 'Wide' is intended for charts. 'Small' shows a compact tile; if the tile is configured to show data, labels are omitted.



Field	Description
Open In	How to run the function: in the main area (replacing the Desktop), in a new window (or tab, depending on the behaviour of your browser), or in a frame on top of the Desktop.
Foreground	<p>The colour of the text on the tile. You can use the colour picker to choose a colour, or you can enter a hexadecimal colour code directly.</p>  <p>If this is blank, the text is black.</p>
Background	The colour of the tile background. As with the foreground, you can use the colour picker or enter a hexadecimal code. If this is blank, the tile background is white.
Icon	The icon to display on the tile. If this is blank, the default icon, which is based on the category of the function, is shown on the tile.
Options	
Values	If the function has been written to allow data to be displayed on the tile, this allows you to select which data to show.

For our first tile, enter the title of Report Status Workbench, Open In Frame and select the values for Waiting, Generating and Unread.

Save the entry and test the Desktop.



Where values are displayed, such as our tile above, the polling of the Tile Values (refresh) is controlled by the same parameter that controls the User Info Messages on Report Status etc and this param is SY/USERINFO



### 10.1.10 Info Tiles

In the previous example, we used Tiles to link to functions that allowed us to work with data in COINS.

Another type of Tile, called an Info Tile, can be used to simply display information. This is essentially a detail page presented as a tile.

It is important to note that Info Tiles do not automatically refresh – the data displayed will remain static until a manual refresh of the desktop is done. For this reason, they may not be suitable on a dashboard screen where dynamic data is required.

Create a new function called SYALPHAINF1. For this to work correctly, it is IMPORTANT that the Category used is INF – Information Tile and the program is set as wou005.

The description for now can be simply Information.

Function	Description	Type	Module	Category	Program	Access Type
SYALPHA	SYALPHA Desktop	Menu	SY-System	MEN-Menu		Read Only
SYALPHAT1	System	Desktop Tab	SY-System	MEN-Menu		Read Only
SYALPHAT1S1	Utilities	Desktop Section	SY-System	MEN-Menu		Read Only
SYALPHAT1S2	Maintenance	Desktop Section	SY-System	MEN-Menu		Read Only
SYALPHAT2	Modules	Desktop Tab	SY-System	MEN-Menu		Read Only
SYALPHAT2S1	Purchase Ledger	Desktop Section	SY-System	MEN-Menu		Read Only
SYALPHAT2S2	Procurement	Desktop Section	SY-System	MEN-Menu		Read Only
SYALPHAINF1	Information	Function	SY-System	INF-Information Tile	wou005	Read Only

Context: Information

Program:

Parameters:

Notes:

Parent:

Access Type:

Role Type: -

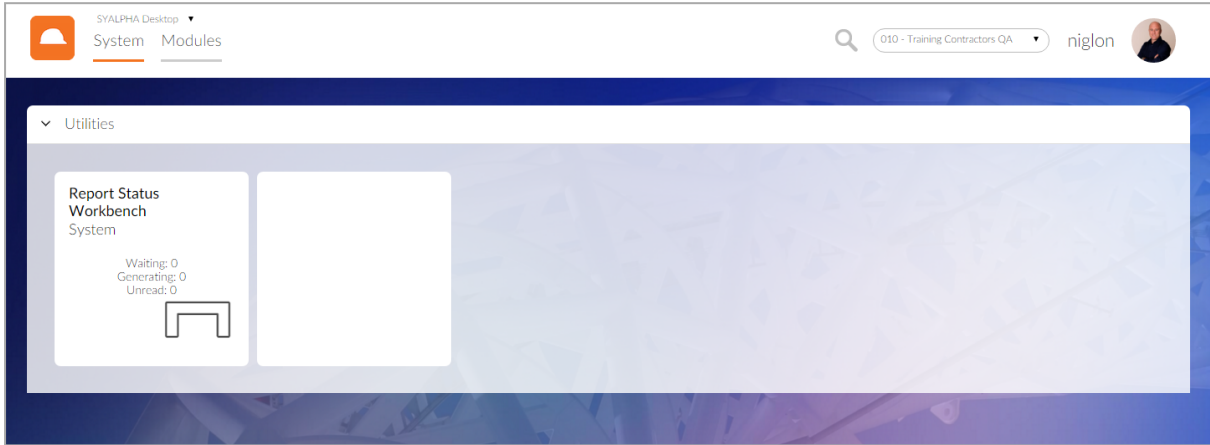
Save this new function, and then add it to the Menu Tab SYALPHAT1S1, setting the size to Normal.

Function	Description	Title	Override	Subtitle	Size	Open In	Type	Module	Category	Row	Column
WWSYBCRR	Report Status	Report Status Workbench	<input type="checkbox"/>		Normal	Main Area	Container	SY-System	Workbench		
SYALPHAINF1	Information		<input type="checkbox"/>		Normal	Frame	Function	SY-System	Information Tile		



10 Building COINS Desktops - Overview

Refresh the Desktop to see the result – you should now have an empty Info Tile displayed on the Utilities section.



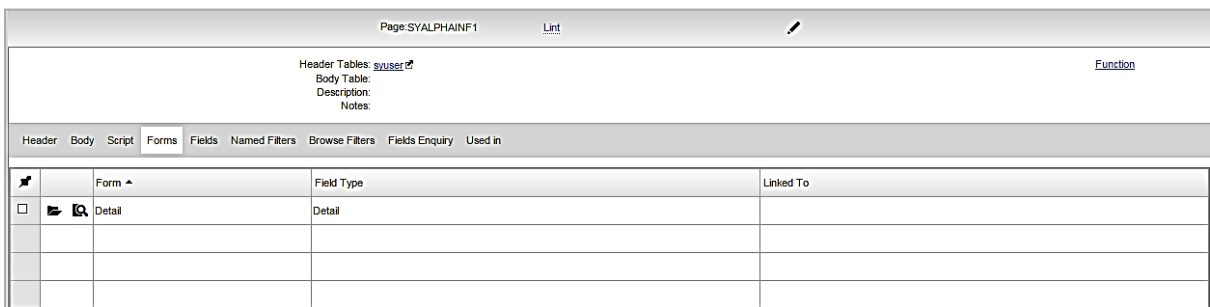
Now we need to define the Detail Page that will populate the tile.

By default, when a user logs into COINS, tables such as co\_config, sysuser and menuparm are already available, so for our first Info Tile we will build a simple tile that displays the details of the logged in user.

In Page Designer, create a new Page Section also called SYALPHAINF1.

Set the Header Table as sysuser.

Click Save and add a Detail Form to the Page.



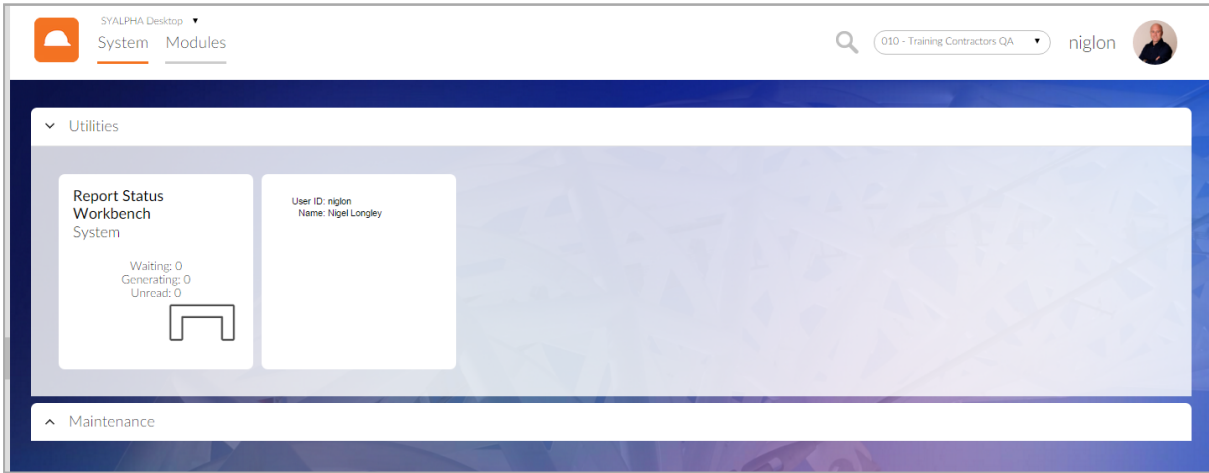
On the Fields Tab, set the Form filter to Detail and apply the filter.

Now on the Details Form add the fields su-userid and su-name



Header Tables: <a href="#">sysuser</a> <span style="float: right;"><a href="#">Function</a></span>										
Body Table: Description: Notes:										
Header	Body	Script	Forms	Fields	Named Filters	Browse Filters	Fields Enquiry	Used in		
Field	Label	Width	Height	Function	Add	Upd	View As	Tab		
<input type="checkbox"/> su-userid	User ID	0	0		<input type="checkbox"/>	<input type="checkbox"/>				
<input type="checkbox"/> su-name	Name	0	0		<input type="checkbox"/>	<input type="checkbox"/>				

Again, refresh the Desktop to see the changes.



All of the functionality of a standard Detail Page are available to be used on Info Tiles.

To demonstrate this we will add a layout to the data.

Create a new field on the Detail Form call LAY1, with the Label of User Details and set View AS to Layout. In the Class, enter LAYA.

Header Tables: <a href="#">sysuser</a> <span style="float: right;"><a href="#">Function</a></span>										
Body Table: Description: Notes:										
Header	Body	Script	Forms	Fields	Named Filters	Browse Filters	Fields Enquiry	Used in		
Field	Label	Width	Height	Function	Add	Upd	View As	Tab	Layout	Append Hidden
<input type="checkbox"/> LAY1	User Details	0	0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Layout		<input type="checkbox"/>	<input type="checkbox"/>




  

Layout	Scripts
Span Label: <input type="text"/> Label Column Spans: <input type="text"/> 0 Column Spans: <input type="text"/> 0 Row Spans: <input type="text"/> 0 Mandatory: <input type="checkbox"/> No Break Label: <input type="checkbox"/> Show in Help: <input checked="" type="checkbox"/> Alignment: <input checked="" type="radio"/> Default <input type="radio"/> Left <input type="radio"/> Centre <input type="radio"/> Right Format: <input type="text"/> Class: LAYA Label Class: <input type="text"/> Build: <input type="text"/> Generate: <input type="text"/>	OnBlur: <input type="text"/> Search   Replace all 1 onChange: <input type="text"/> Search   Replace all 1

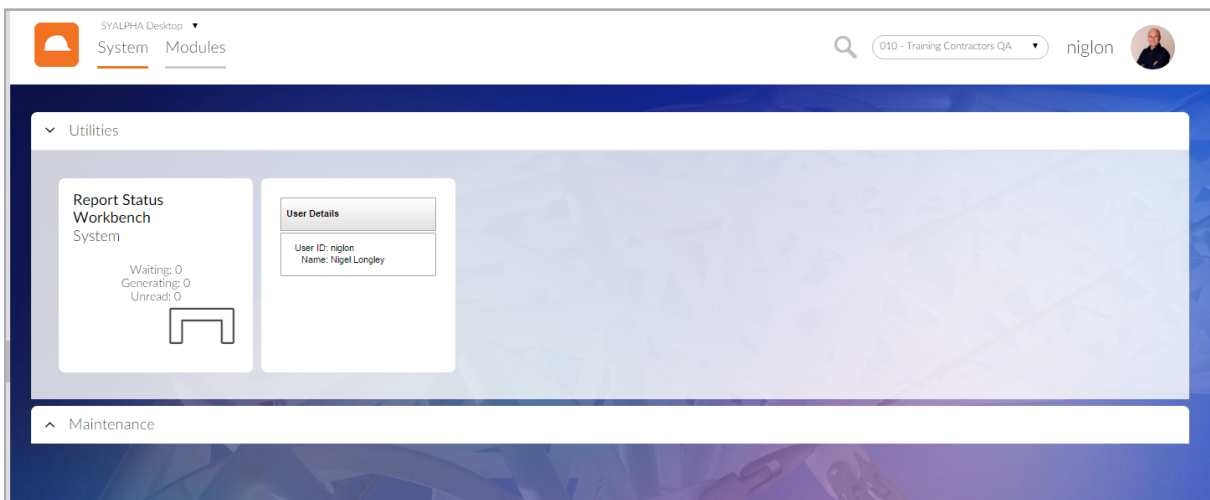




Save the new field and then for su-userid and su-name, set the Layout field to LAY1.

Page: SYALPHAINF1 <a href="#">Lint</a>												
Header Tables: <a href="#">sysuser</a>											Function	
Body Table:												
Description:												
Notes:												
Header	Body	Script	Forms	Fields	Named Filters	Browse Filters	Fields Enquiry	Used in				
Field	Label	Width	Height	Function	Add	Upd	View As	Tab	Layout	Append	Hidden	
<input type="checkbox"/>  su-userid	User ID	0	0		<input type="checkbox"/>	<input type="checkbox"/>			LAY1	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>  su-name	Name	0	0		<input type="checkbox"/>	<input type="checkbox"/>			LAY1	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>  LAY1	User Details	0	0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Layout			<input type="checkbox"/>	<input type="checkbox"/>	

Refresh the Desktop to see the changes.



The screenshot shows the SYALPHA Desktop interface. At the top, there is a navigation bar with 'System' and 'Modules' tabs, a search bar containing '010 - Training Contractors QA', and a user profile for 'niglon'. Below the navigation bar, the 'Utilities' section is expanded, showing two widgets: 'Report Status Workbench System' and 'User Details'. The 'Report Status Workbench System' widget displays statistics: 'Waiting: 0', 'Generating: 0', and 'Unread: 0'. The 'User Details' widget shows 'User ID: niglon' and 'Name: Nigel Longley'. At the bottom of the interface, there is a 'Maintenance' section which is currently collapsed.



## 10.1.11 Using Calculation Programs to prepare data for the Desktop

Our Info Tile was able to access information from co\_config, sysuser and menuparm without any additional work on our part.

It is possible to set up access to other tables before the desktop loads using Calculation Programs, this will then allow the tiles access to that information..

Calculation programs are essentially pre-written standard OA Calculation syntax that can be called on reports and browse screens. They can be as complex as required.

For example, for House Sales there is a standard Calculation program called %FINDVSI

System   Calculation Programs		10 - Training Contractors QA   nignlon	COINS
Program ^	Description		
%FINDVSI	Find Desktop Development Record		
Notes: Smethod\$('vsi-rsp.findKey',co_config.ro_varvalue^kco,co_config.ro_varvalue^pvJob,co_config.ro_varvalue^pvPhase);			

This uses the following method to access vsi-rsp with a given kco, Development and Phase code to return the RowID of that Development.

```
$method$('vsi-rsp.findKey',co_config.ro_varvalue^kco,co_config.ro_varvalue^pvJob,co_config.ro_varvalue^pvPhase);
```

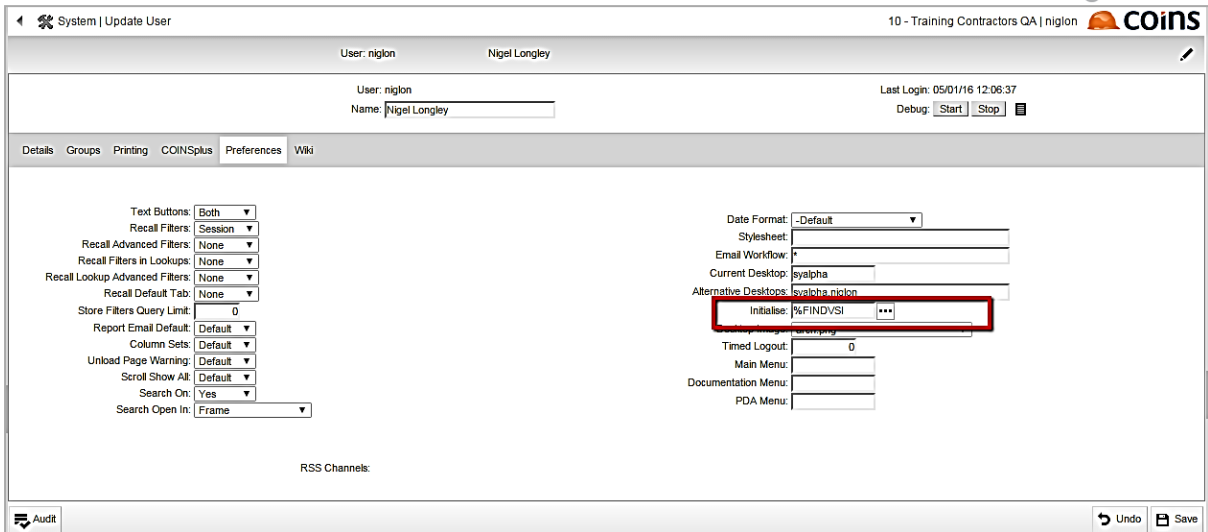
Note that the values are stored as kco (company number retrieved from the URL), pvJob and pvPhase. The pv prefix indicates that these are permanent variables and once stored are retained in the URL until removed or their value changes. Setting the pv values allows a specific development to be selected and any relevant Info Tiles on the next Desktop build will then show information for that development.

Currently (Jan 2016) there are only four pv values currently available – JC job (pvJob) and phase (pvPhase) and CI (pvView) and SRM (pvSibling).

To see how this can work, first we need to add this program to our user record.

In User Maintenance, find your userid and open the record. Select the Preferences Tab.

In the Initialise field, enter %FINDVSI

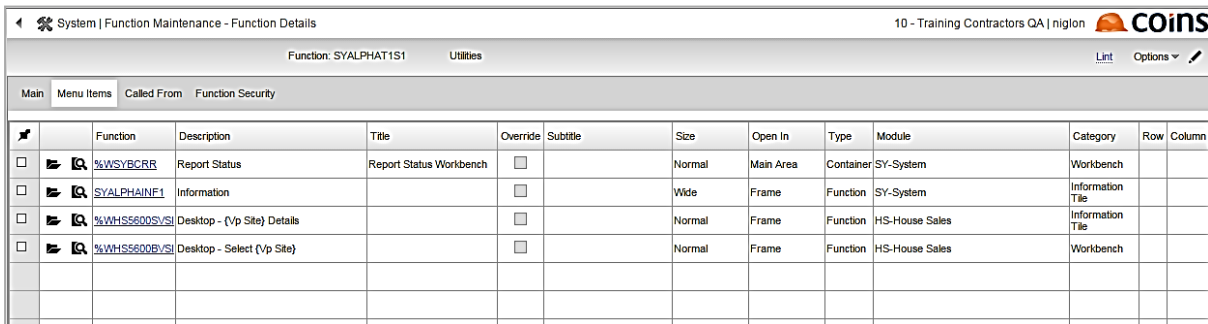
Save the record.

For demonstration purposes we are going to borrow some functionality from the Standard House Sales Dashboard

On the Utilities section of your Desktop add the two following Coins Standard Function as separate entries.

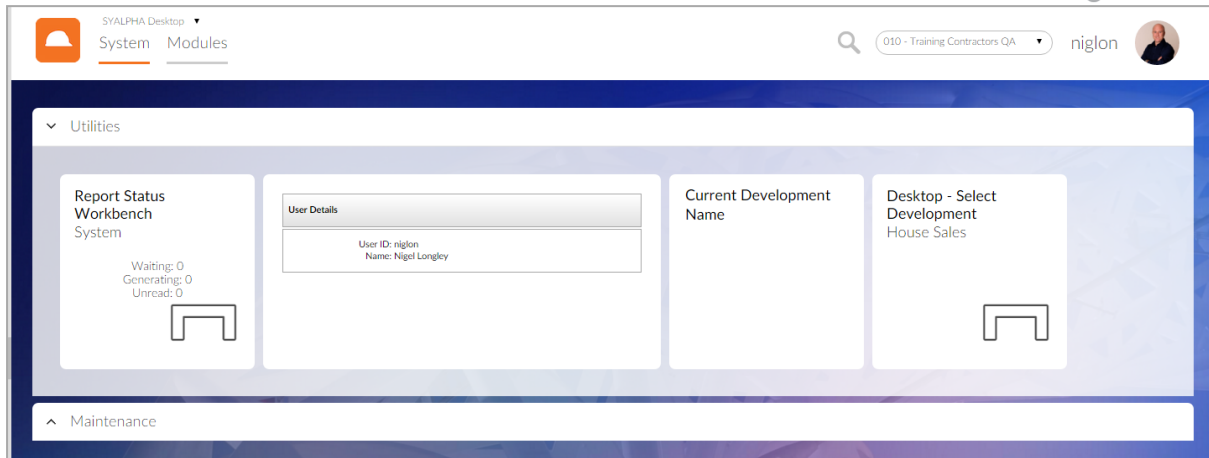
%WHS5600SVSI

%WHS5600BVSI



Function	Description	Title	Override	Subtitle	Size	Open In	Type	Module	Category	Row	Column
%WSYBCRR	Report Status	Report Status Workbench	<input type="checkbox"/>		Normal	Main Area	Container	SY-System	Workbench		
SYALPHAINF1	Information		<input type="checkbox"/>		Wide	Frame	Function	SY-System	Information Tile		
%WHS5600SVSI	Desktop - (Vp Site) Details		<input type="checkbox"/>		Normal	Frame	Function	HS-House Sales	Information Tile		
%WHS5600BVSI	Desktop - Select (Vp Site)		<input type="checkbox"/>		Normal	Frame	Function	HS-House Sales	Workbench		

Now refresh the Desktop.



On first run, the current development Tile will be blank as it does not know what information we want. This is not good practice as in a more complex desktop, tiles that cannot retrieve their information may cause the desktop to fail, so your desktop would normally be configured to populate with something on first load

Click the Select Development Tile, this runs a function that allows a development to be selected, runs the calculation program and rebuilds the desktop. Choose a development (this guide assumes you are using a COINS training environment configured for House Sales)

The Info Tile should now change to reflect the chosen development. In the URL note that the pv Values are shown. Any Info Tile can use those values to reflect information relating to the selected Development. These pv Values will be retained in the URL until another development is selected.



### 10.1.12 Desktop Tile Values

Certain Tile Values are hard-coded against a function – such as the ones we selected for our first Tile. Typically these will be for values that are standard for any COINS user, such as the number of report generating, the number of Uncommitted Orders, the number of users on the system etc.

COINS functions that have been set up to display tile values have an information code specified in the function parameters (for example: &info=cosval .reports on the Report Status Workbench function).

Function	Description	Type	Module	Category
%WSYBCRR	Report Status	Container	SY-System	WBN-Workbench

<p>Context: Report Status                  Program: wou005                  Parameters: info=cosval.reports                  Notes:                  Parent:                  Access Type:                  Role Type: SYS-System</p>
---

For specific user requirements, it is possible to define other values which can be used on tiles.

You can either set up additional calculations that are available to any function that uses the same information code, or set up calculations that will be available on any tile by leaving the information code blank.

Before writing a new Desktop Tile Value to address a specific client’s needs, consideration should be given as to whether Development should hard code the calculation against the function or if a bespoke Tile value is needed.

There may be performance implications – see notes (in red) below

#### 10.1.12.1 Examples of Desktop Tile Values

Desktop Tile Values allows you to set up calculations to show information on tiles on the COINS Desktop.

#### Accessing a Table Field



System - Desktop Tile Values 20 - Housebuilders QA - Nigel Longley

Info Code	Info ID	Description
FUNCTION.%WHSBVWB	%AVAIL	Available Plots

Configuration Tables:   
 Calculation:

```
1 iAvailPlots = vp_site.RO_hds_avail;
```

### Accessing a Data Mart Value

System - Desktop Tile Values 20 - Housebuilders QA - Nigel Longley

Info Code	Info ID	Description
FUNCTION.%WHS5604BVWB	%COMPTODAY	Exchanges Expected Today

Configuration Tables: **co\_config**   
 Calculation:

```
1 $param = "%HSDTOP02|LAST|KCOJOBETD|dwa_count|" + vp_site.kco + '|' + vp_site.job_num + '|1|1|1|1|1|0';
2 iCompOdue = method('coc-rsp.getRO_coc_dmvalue',param);
```

### Using a Method to Write a Query to Retrieve Information

System - Desktop Tile Values 20 - Housebuilders QA - Nigel Longley

Info Code	Info ID	Description
	PO_OSORDERS	Open Orders

Configuration Tables: **co\_config**   
 Calculation:

```
1 method('syuquery.aggregate',"for each po_hdr where kco = " + co_config.kco + " and po_hdr.tip_type = 'MATERIAL' and po_hdr.poh_accno = " + co_config.RO_diaryUser + "' AND poh_chgno = " + co_config.kco");
2 method('syuquery.getDAgg',1);
3 method('syuquery.getIAgg');
```

Field	Description
-------	-------------



Info Code	<p>The information code to identify which function(s) to make this value available on.</p> <p>COINS functions that have been set up to display tile values already have an information code specified in the function parameters (for example: &amp;info= cosval .batches). If a user-defined value has an information code that matches the code on the function (for example, cosval.batches), the user-defined value will be available for users to show on the tile, in addition to the standard values.</p> <p>The information code can also be FUNCTION.&lt;MainArea&gt; where FUNCTION indicates that this value is for a single function only and &lt;MainArea&gt; is the function code.</p> <p>If the information code is blank, the user-defined value will be available for users to show on any tile.</p>
Info ID	The information value ID. The combination of info code and info ID must be unique.
Description	A description of the user variable value. This is used as the label for this value on the tile.
Configuration Tables	<p>The configuration tables that should be positioned before the calculation is evaluated.</p> <p>co_config, sysparam and menuparm are available by default, pointing at the correct records.</p>
Calculation	<p>The calculation to evaluate for the value shown on the tile.</p> <p>The calculation can be any standard OA Designer Calculation syntax as in the examples above.</p> <p>Note that the values for all tiles will be updated with a frequency that is determined by the SY/USERINFO parameter. Therefore, although the calculations can be quite complex (using Method Queries for example), if they take too long to run this could have an adverse effect on performance.</p>







### 10.1.1 Tile Options

Desktop Tile Options allows you to set up alternative behaviour for the links on COINS Desktop tiles (for example, to open a specific tab or a named filter). These are then available as options that the user can select when configuring the tile.

In effect, this allows you to have the same function, multiple times, on a Desktop with each tile displaying different information.

Function	Option	Description
%WHSBWB	%AVAIL	Available Plots
%WPO101ABPHL	%UNCOMMITTED	Uncommitted Orders
%WPO101BBPHL	%UNCOMMITTED	Uncommitted Orders
%WSYBCRR	%WSY2110BCRR	Scheduler Tab
%WSYBCRR	%WSY2130BCRR	Report Status Tab

Function	Option	Description
%WHSBWB	%AVAIL	Available Plots
%WPO101ABPHL	%UNCOMMITTED	Uncommitted Orders
Parameters: &queryFilterType=simple&queryColumnFilter=&NamedFilter=UncommittedOrders\OsOnly,AllBuyers&initContainer=%WPO101ATPHL Security Function: Build Condition:		
%WPO101ATPHL	Committed_Mine	My Committed Orders
%WPO101ATPHL	UnCommitted_Mine	My UnCommitted Orders
%WPO101BBPHL	%UNCOMMITTED	Uncommitted Orders
%WSYBCRR	%WSY2110BCRR	Scheduler Tab
%WSYBCRR	%WSY2130BCRR	Report Status Tab



Field	Description
Function	The function to which the tile option should be made available.
Option	The ID of the tile option.
Description	A description of the tile option. This is used as the label for this option in the Options drop-down list on the tile maintenance screen.
Parameters	<p>The parameters that should be passed to the link when the tile is selected.</p> <p>The parameters are a string of values that are set in the URL when the tile is selected. This may typically be :</p> <p>initTab to set a tab,</p> <p>initContainer to set a container tab or</p> <p>NamedFilter to set the named filter on that function to be used (if available). This is a comma separated list of the values to be used.</p> <p>queryFilter to determine the filter type to use (simple or advanced)</p> <p>queryColumnFilter to set the column that you are filtering on</p> <p>The parameters may be used either alone or in combination with '&amp;' as the separator (no spaces).</p> <p>E.g.</p> <p><code>&amp;queryFilterType=simple&amp;NamedFilter=UnCommittedOrdersOnly,MyOrders</code></p>
Security Function	<p>An optional security function associated with the option.</p> <p>Typically this would be used for tabs where the user will only be able to access a tab if the function is available to them and the tile value should therefore also only be available based on this function. If access is not available to the user then the option will be suppressed.</p> <p>If blank then the option will be available to all users.</p>
Build Condition	An optional build condition that determines whether the option is available.



10.1.13 NamedFilter Example

The NamedFilter parameter allows a Tile Option to access any existing Named Filters on a function and to pre-populate (and apply) them with values before building the browse.

NamedFilter affects the URL, so must always be preceded by “&” when used so that it correctly becomes a parameter in the URL.

Desktop Tile Options

%WPO101ATPHL is the Function for the Material Orders Workbench. This function has a two sets of pre-defined Named Filters for the Types of Order (Committed, Uncommitted etc.) and The Buyers (All or My Orders)

Page: %WPO101ATPHL					Limit			
Header Tables: <a href="#">co_confir</a> , <a href="#">po_confir</a> <span style="float: right;">Function</span>								
Body Table: <a href="#">po_hdr</a>								
Description:								
Notes: Material Orders (Live) Column Fields: 2 std:Export - from v11.03 the system uses the last entry of duplicate fields instead of the first one as it is in v11.02 and earlier versions.								
Header	Body	Script	Forms	Fields	Named Filters	Browse Filters	Fields Enquiry	Used In
Name	Table	Build	New					
<input type="checkbox"/> All Orders	po_hdrfive		<input type="checkbox"/>					
<input type="checkbox"/> Exclude Cancelled Orders	po_hdr		<input type="checkbox"/>					
<input type="checkbox"/> Incomplete Orders & VO's Only	po_hdrfive		<input type="checkbox"/>					
<input type="checkbox"/> Incomplete Original Orders Only	po_hdrfive		<input type="checkbox"/>					
<input type="checkbox"/> Un-Released Orders Only	po_hdr	poF-rsp.UsingPOWorkflow	<input type="checkbox"/>					
<input type="checkbox"/> Released Orders Only	po_hdr	poF-rsp.UsingPOWorkflow	<input type="checkbox"/>					
<input type="checkbox"/> Committed Orders Only	po_hdrfive		<input type="checkbox"/>					
<input type="checkbox"/> Uncommitted Orders Only	po_hdrfive		<input type="checkbox"/>					
<input type="checkbox"/> Uncommitted Orders & VO's Only	po_hdr		<input type="checkbox"/>					
<input type="checkbox"/> Foreign Currency Orders Only	po_hdrfive		<input type="checkbox"/>					
<input type="checkbox"/> Cancelled Orders Only	po_hdr		<input type="checkbox"/>					
<input type="checkbox"/> Head Office Orders Only	po_hdrfive		<input type="checkbox"/>					
<input type="checkbox"/> Site Orders Only	po_hdrfive		<input type="checkbox"/>					
<input type="checkbox"/> Orders Requiring Call-Offs	po_hdrfive	poF-rsp.buildSysCallOffs	<input type="checkbox"/>					
<input type="checkbox"/> Reserved Orders Only	po_hdrfive		<input type="checkbox"/>					
<input type="checkbox"/> All Templates	po_hdrfive		<input type="checkbox"/>					
<input type="checkbox"/> Unassigned Templates	po_hdrfive		<input type="checkbox"/>					
<input type="checkbox"/> Replan (PO Variation) Orders	po_hdr		<input type="checkbox"/>					
<input type="checkbox"/> All Buyers	po_hdrfive		<input checked="" type="checkbox"/>					
<input type="checkbox"/> My Orders	po_hdrfive		<input type="checkbox"/>					

On my desktop I want to have three tiles, one for All Orders, one for My Committed Orders and one for my uncommitted orders. Running the function alone should give me All orders by default (subject to my filter recall settings). So I only need to set up two Tile Value Options against %WPO101ATPHL.



System   Desktop Tile Options		10 - Training Contractors QA   nigon	
Function	Option	Description	
%WPO101ATPHL	Committed_Mine	My Committed Orders	
Parameters: &NamedFilter=CommittedOrdersOnly,MyOrders Security Function: Build Condition:			
%WPO101ATPHL	UnCommitted_Mine	My UnCommitted Orders	
Parameters: &NamedFilter=UnCommittedOrdersOnly,MyOrders Security Function: Build Condition:			

For the first, the parameter will be:

&NamedFilter=CommittedOrdersOnly,MyOrders

This sets the First Named Filter to Committed Orders Only and the second filter to My Orders.


The second function entry has a similar parameter, but sets the first filter to UnCommitted Orders.

**Desktop**

On our Example Desktop %WPO101ATPHL has three separate Tiles defined.

- For the first entry, the Title is set to All Orders and the Option is left as default.
- For the second Entry the Title is set to My Committed Orders and the Option is set to My Committed Orders.
- For the third Entry the Title is My Uncommitted Orders, the Option is set to My Uncommitted orders




System | Function Maintenance - Function Details 10 - Training Contractors QA | nignlon 

Function: SYALPHAT2S2 Procurement [Link](#) Options

Main Menu Items Called From Function Security

Function	Description	Title	Override	Subtitle	Size	Open In	Type	Module	Category	Row	Column
%WPO101ATPHL	Material Orders	All Orders	<input type="checkbox"/>		Small	Frame	Tab	PO-Purchase Orders	Workbench		
Foreground: <input type="checkbox"/> Background: <input type="checkbox"/> Icon: <input type="checkbox"/> Icon: <i class="fa-2x ci" style="width:40px;>&nbsp;</i> Options: Default Values:											
%WPO101ATPHL	Material Orders	My Committed Material Orders	<input type="checkbox"/>		Small	Frame	Tab	PO-Purchase Orders	Workbench		
Foreground: <input type="checkbox"/> Background: <input type="checkbox"/> Icon: <input type="checkbox"/> Icon: <i class="fa-2x ci" style="width:40px;>&nbsp;</i> Options: My Committed Orders Values:											
%WPO101ATPHL	Material Orders	My UnCommitted Material Orders	<input type="checkbox"/>		Small	Frame	Tab	PO-Purchase Orders	Workbench		
Foreground: <input type="checkbox"/> Background: <input type="checkbox"/> Icon: <input type="checkbox"/> Icon: <i class="fa-2x ci" style="width:40px;>&nbsp;</i> Options: My UnCommitted Orders Values:											

The refreshed Desktop now looks like this:

SYALPHA Desktop 010 - Training Contractors QA | nignlon 

System Modules

Purchase Ledger

Procurement

All Orders  
Purchase  
Orders

My Committed  
Material Orders  
Purchase  
Orders

My UnCommitted  
Material Orders  
Purchase  
Orders

Clicking the first tile we get all orders and all buyers

All Orders

Material Orders By Contract

PO Number	Chg No	Supplier	Account	Original Order Date	Order Date	Contract	Status	Com	Buyer	Type	Amount	Cur
8777		Prater Roofing	PRA001	04/05/10	04/05/10	FM57	COM	<input type="checkbox"/>	000500	N	9,546.00	EUR
8888		Edmundson Electrical Ltd	EDM001	04/05/10	04/05/10	FM57		<input type="checkbox"/>	000500	N	0.00	GBP
8889		Prater Roofing	PRA001	04/05/10	04/05/10	FM57	COM	<input type="checkbox"/>	000500	N	9,546.00	EUR
Cement		Edmundson Electrical Ltd	EDM001	14/04/10	14/04/10	FM57		<input type="checkbox"/>	ewahug	T	0.00	GBP
cement_2		Edmundson Electrical Ltd	EDM001	16/04/10	16/04/10			<input type="checkbox"/>	ewahug	T	0.00	GBP
HM-100000001		Fast Delivery Specialist	FDS001	20/05/11	20/05/11	10000		<input type="checkbox"/>	karmas	N	28,043.00	GBP

Filter: All Orders | All Buyers Search: PO Number

The second tile will automatically give us Committed orders, my orders



My Committed Material Orders

PO Number	Chg No	Supplier	Account	Original Order Date	Order Date	Contract	Status	Com	Buyer	Type	Amount	Cur
<a href="#">HM-10000/0004</a>		Abbey Glass	ABB001	30/09/11	30/09/11	10000	COM	<input type="checkbox"/>	niglon	N	1,000.00	GBP
<a href="#">HM-10000/0005</a>		Abbey Glass	ABB001	30/09/11	30/09/11	10000	COM	<input type="checkbox"/>	niglon	N	1,000.00	GBP
<a href="#">HM-10000/0007</a>		Abbey Glass	ABB001	01/01/11	01/01/11	10000	COM	<input type="checkbox"/>	niglon	N	1,500.00	GBP
<a href="#">HM-10000/0011</a>		Jewson Limited	JEW001	24/06/15	24/06/15	10000	COM	<input type="checkbox"/>	niglon	N	1,250.00	GBP
<a href="#">HM-10000/0012</a>		Abbey Glass	ABB001	06/07/15	06/07/15	10000	COM	<input type="checkbox"/>	niglon	N	1,500.00	GBP
<a href="#">HM-21100/0002</a>		AG Lawrence (Plastering) Ltd	AGLA005	16/05/12	16/05/12	21100	COM	<input type="checkbox"/>	niglon	N	405.00	GBP

Filter: Committed Orders Only | My Orders | Search: PO Number

The final tile will automatically display Uncommitted Orders, My Orders

My UnCommitted Material Orders

PO Number	Chg No	Supplier	Account	Original Order Date	Order Date	Contract	Status	Com	Buyer	Type	Amount	Cur
<a href="#">HM-10000/0006</a>		Abbey Glass	ABB001	01/01/11	01/01/11	10000		<input type="checkbox"/>	niglon	N	1,000.00	GBP
<a href="#">HM-10000/0008</a>		Abbey Glass	ABB001	01/01/11	01/01/11	10000		<input type="checkbox"/>	niglon	N	30,000.00	GBP
<a href="#">HM-10000/0010</a>		Jewson Limited	JEW001	01/06/15	01/06/15	10000	REL	<input type="checkbox"/>	niglon	N	1,250.00	GBP

Filter: Uncommitted Orders Only | My Orders | Search: PO Number



10 Building COINS Desktops - Overview

10.1.13 InitContainer Example

On Container Functions with multiple tabs (For example the Report Status Workbench) the parameter InitContainer allows the tile to preselect the tab to display when the tile is clicked.

Desktop Tile Options

The Activity Workbench (%WHS1150SHSAI) is a Container Function

Function	Description	Type	Module	Category	Program	Access Type
%WHS1129CHST	Journal Lookup	Container	HS-House Sales	ENQ-Enquiry	wou005	Read Only
%WHS1150SHSA	Activity Workbench	Container	SY-System	WBN-Workbench	wou005.p	Read Only

It has the following tabs:

Function	Description	Context	Type	Module	Category
%WHS1153BHSA	Todays Actions		Function	SY-System	Workbench
%WHS1154BHSA	My New Actions		Function	SY-System	Workbench
%WHS1151BHSA	Appointments		Function	SY-System	Workbench
%WHS1152BHSA	Tasks		Function	SY-System	Workbench
%WHS1150BHSA	All Actions		Function	SY-System	Workbench
%WHS1160BHSA	My (Diary)		Function	SY-System	Maintenance
%WHS1161BHSA	Group View		Function	SY-System	Maintenance
%WHS1150BSUR	Another User View		Function	SY-System	Maintenance

To create a Tile that will take us directly to the My New Actions Tab we can set up a Desktop Tile Value as below:

Function	Option	Description
%WHS1150SHSA	%WHS1154BHSA	My New Actions

Parameters:

Security Function: %WHS1150SHSA

Built Condition:

Desktop

If we add the Activity Workbench function to our desktop, the Option field will now display the options of Default (use the standard recall setting on my user preferences) or My New Actions.



System | Function Maintenance - Function Details 20 - Housebuilders QA | nigon

Function: SYALPHAT1S1 Utilities Lint Options

Main Menu Items Called From Function Security

Function	Description	Title	Override	Subtitle	Size	Open In	Type	Module	Category	Row	Column
%WSYBCRR	Report Status	Report Status Workbench	<input type="checkbox"/>		Normal	Main Area	Container	SY-System	Workbench		
SYALPHAINF1	Information		<input type="checkbox"/>		Wide	Frame	Function	SY-System	Information Tile		
%WHSS600S/VS	Desktop - (Vp Site) Details		<input type="checkbox"/>		Normal	Frame	Function	HS-House Sales	Information Tile		
%WHSS600B/VS	Desktop - Select (Vp Site)		<input type="checkbox"/>		Normal	Frame	Function	HS-House Sales	Workbench		
%WHS1150SHS	Activity Workbench		<input type="checkbox"/>		Small	Frame	Container	SY-System	Workbench		

Foreground:

Background:

Icon:

Options: My New Actions

Values: Available:  Selected:

New Outstanding:

Select My New Actions, save and then refresh the desktop.

Now when the Activity Workbench Tile is clicked, it will always open on the My New Actions Tab

Activity Workbench ? ☰ - e x

Todays Actions My New Actions Appointments Tasks All Actions My Diary Group View Another User View

Cat	Date	Time	Type	Subject	Co	Link	Done
Task	01/06/15	16:34	TASK-Task	Workflow Exercise 2 - Approve Order	010	<a href="#">Link</a>	<input type="checkbox"/>

Appointment

Filter: To Do Search: Cat





### 10.1.1 Tables on Tiles

Tables are scrollable tables of information that can be used anywhere, but here we concentrate on adding them to a Tile.

In order to put a table on a tile, you need to build a browse with fields on it. This can be directly against a database table, if you already have data or you can build a dataset to create the data required before the chart is produced.

Tables offer the benefit of a pre-built search field, allowing any information in the tile to be located, and (since the data is sent client side) the ability to include RO\_Fields on the table which will be sortable – something that previously required a dataset.



10.1.14 Tables : Example 1 – Direct Query against the database

In this example, we will use the table ap\_vendbal to display a table of supplier period balances

**Create the Function**

The function is a standard function with the exception that the Category must be TBL – Table. Be careful not to select Category TAB – DB Table by mistake

Function	Description	Type	Module	Category	Program	Access Type
SYALPHAAVPAY	Supplier Balances	Function	SY-System	TBL-Table	wou005	Read Only

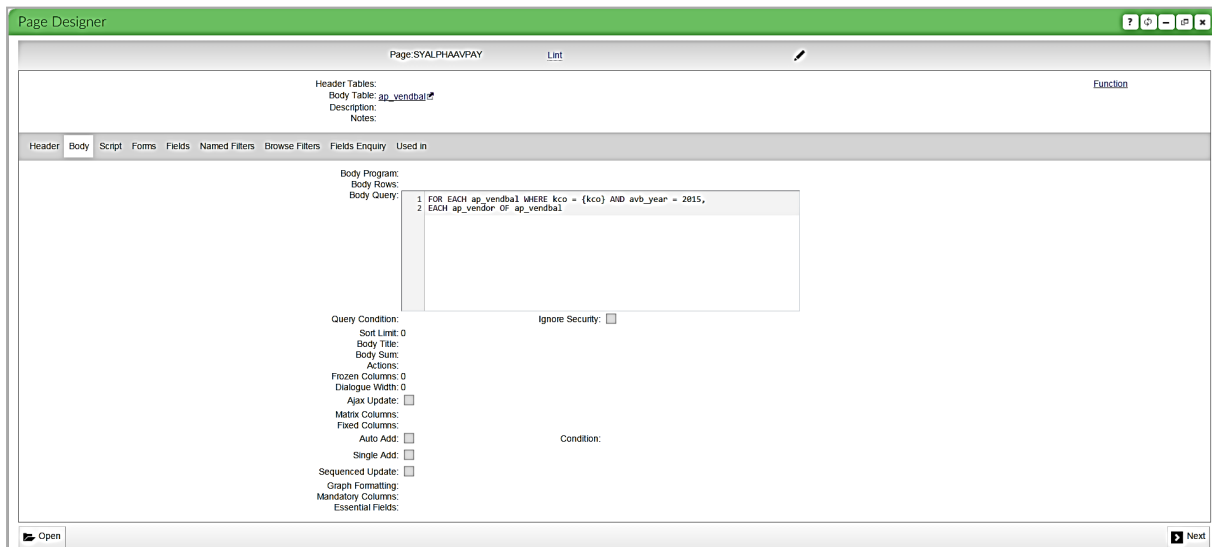
  

Context: Supplier Balances  
 Program: wou005  
 Parameters:  
 Notes:  
 Parent: SYALPHAAVPAY  
 Access Type: Read Only  
 Role Type: -

**Create the Page**

The Page should use the table ap\_vendbal as its body table and with an appropriate body query to retrieve the records to be displayed. For Table Tiles DO NOT specify a number of rows.

When creating a table tile, give consideration to the number of rows that will be returned. Since the data is all sent client side there will be a performance issue if a tile returns too much information across multiple desktops. A maximum of 1000 records should be considered best practice.





10Building COINS Desktops - Overview

On our body fields, we need to specify just the fields we are displaying.

For each value, you must specify an ID (this can be any unique reference for each value)

Field	Label	Width	Height	Function	Add	Upd	View As	Tab	Layout	Append	Hidden
avm_num	A/C No	5	0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>
avm_name	Supplier	5	0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>
avb_pbal_1	January	10	0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>
avb_pbal_2	February	10	0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>
avb_pbal_3	March	10	0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>
avb_pbal_4	April	10	0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>
avb_pbal_5	May	10	0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>
ap_vendbal_RO_avb_total*B	Balance YTD	10	0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>

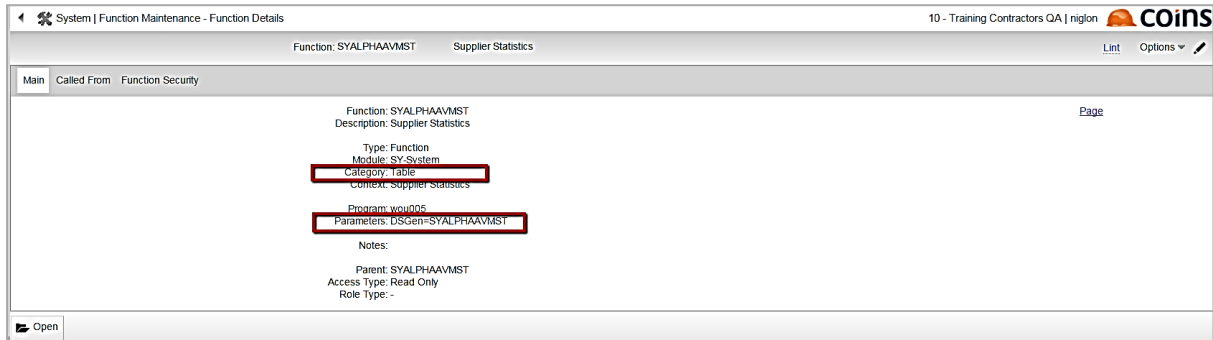
Desktop

Adding the Function to a desktop will produce the following output (Tile has been set to Full as its size):

A/C No	Supplier	January	February	March	April	May	Balance YTD
ABB001	Abbey Glass	4,222.89	4,222.89	4,222.89	4,222.89	4,222.89	4,222.89
BET001	Better Bricks	43,183.80	43,183.80	43,183.80	43,183.80	43,183.80	43,183.80
BRK005	W A Brick & Stone Ltd	2,400.00	2,400.00	2,400.00	2,400.00	2,400.00	2,400.00
CRY001	Crystal Cleaning	270.00	270.00	270.00	270.00	270.00	270.00
JEW001	Jewson Limited	14,359.00	14,359.00	14,359.00	14,359.00	14,359.00	14,359.00

## 10.1.14 Example 2 – Using a Data Set

In this (and the next example – charts – we will use a dataset to build our data)

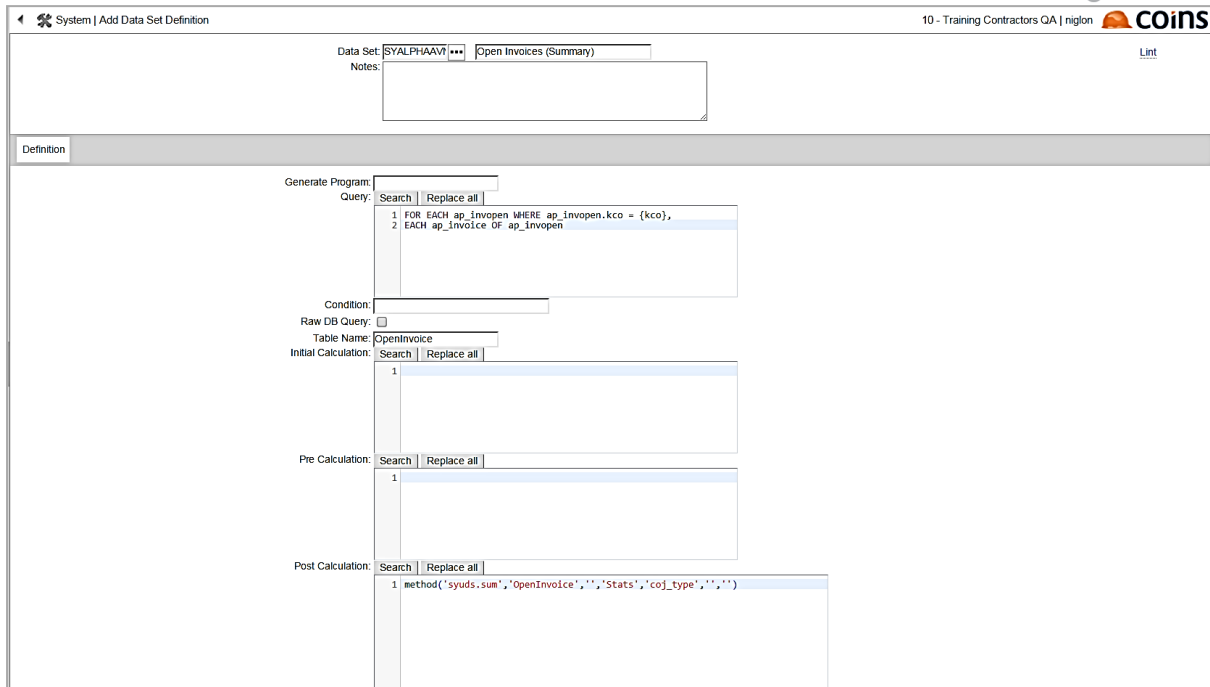
**Create the Function**

The function is a standard function with the exception that the Category must be TBL – Table. Be careful not to select Category TAB – DB Table by mistake

In the parameters field the parameter DSGen= specifies the name of the dataset to be built before the chart is generated.

**Create the Dataset**

Our example dataset will go through all the open invoices for the currently selected company and will then create a summary of the data to be displayed.



In this example, the query will return a large number of invoices, which we could display, but in this instance the Post Calculation is specified to use the Method syuds.sum to aggregate all the returned records by Transaction type so that we have one total value for Gross for each transaction type.

Syuds.sum requires the current temp table (OpenInvoice), a WHERE condition (if required), the new temp table to be created (Stats), the field(s) to aggregate the data by (coj\_type). The remaining to parameter are left blank in this example.

Syuds. Calculation methods are covered in detail in the Guide LMDSY0036 - How to Create and Use Datasets available from the Learning Resources section of the COINS Client Area on our website ([www.coins-global.com](http://www.coins-global.com)) under Business Intelligence and Analytics/OA Designer.

The data set has the following fields defined against it.



System | Data Set Summary 10 - Training Contractors QA | nigel

Data Set: SYALPHA VMST Open Invoices (Summary) [Lint](#)

Notes:

Definition		Fields						
Field	Key	Extent	Label	Data Type	Format	Source		
avm_num	<input type="checkbox"/>	<input type="checkbox"/>	Supplier	Character	x(8)	ap_invoice.avm_num		
ain_inv	<input type="checkbox"/>	<input type="checkbox"/>	Invoice	Character	x(8)	ap_invoice.ain_inv		
coj_type	<input type="checkbox"/>	<input type="checkbox"/>	Type	Character	x(8)	ap_invoice.coj_type		
dGross	<input type="checkbox"/>	<input type="checkbox"/>	Gross	Decimal	->>>.>>>.>>9.99	ap_invoice.ain_amount		

Advanced

**Desktop**

Adding the Function to a desktop will produce the following output:

SYALPHA Desktop

System Modules

010 - Training Contractors QA | nigel

Purchase Ledger

**Invoices**  
Purchase Ledger

282,175

Overdue

**Supplier Stats**  
System

Search:

Type	Gross
PCRN	1,065.50
PINN	15,281.89
PINV	265,892.42

Showing 1 to 3 of 3 entries

Procurement



## 10.1.15 Charts on Tiles

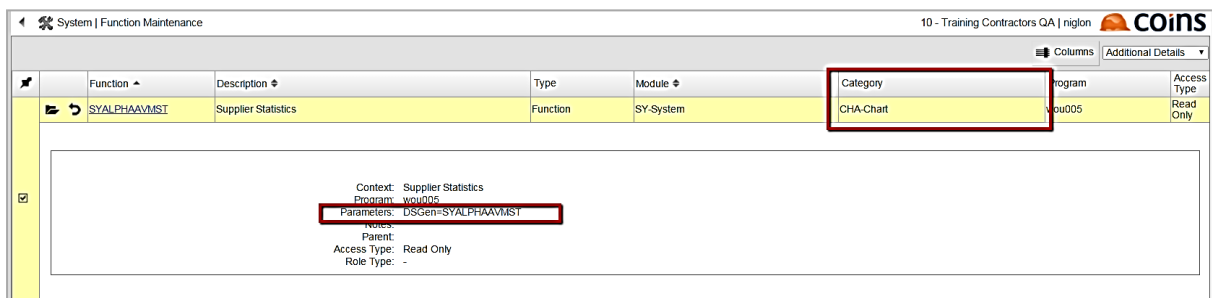
In order to put a chart on a tile, you need to have a set of records with an X coordinate value and a Y coordinate value that you intend to plot.

To achieve this, you need to build a browse with fields on it. This can be directly against a database table, if you already have data to plot or you can build a dataset to create the data required before the chart is produced.

### 10.1.15.1 Dataset driven Charts

In this example, we will create a dataset to generate Supplier Statistics add the chart to a new tile.

#### Create the Function



The function is a standard function with the exception that the Category must be Chart.

In the parameters field the parameter DSGen= specifies the name of the dataset to be built before the chart is generated.

#### Create the Dataset

Our example dataset will go through all the open invoices for the currently selected company and will then create a summary of the data to be displayed.



In this example, the query will return a large number of invoices, which we could plot, but the graph would be very confusing. So in this instance the Post Calculation is specified to use the Method `syuds.sum` to aggregate all the returned records by Transaction type so that we have one total value for Gross for each transaction type.

`Syuds.sum` requires the current temp table (`OpenInvoice`), a `WHERE` condition (if required), the new temp table to be created (`Stats`), the field(s) to aggregate the data by (`coj_type`). The remaining to parameter are left blank in this example.

`Syuds`. Calculation methods are covered in detail in the Guide `LMDSY0036 - How to Create and Use Datasets` available from the Learning Resources section of the COINS Client Area on our website ([www.coins-global.com](http://www.coins-global.com)) under Business Intelligence and Analytics/OA Designer.

The data set has the following fields defined against it.





System | Data Set Summary 10 - Training Contractors QA | nigelon

Data Set: SYALPHAAMVST Open Invoices (Summary) [Lint](#)

Notes:

Definition		Fields						
	Field	Key	Extent	Label	Data Type	Format	Source	
<input type="checkbox"/>	avm_num	<input type="checkbox"/>		Supplier	Character	x(8)	ap_invoice.avm_num	
<input type="checkbox"/>	ain_inv	<input type="checkbox"/>		Invoice	Character	x(8)	ap_invoice.ain_inv	
<input type="checkbox"/>	coj_type	<input type="checkbox"/>		Type	Character	x(8)	ap_invoice.coj_type	
<input type="checkbox"/>	dGross	<input type="checkbox"/>		Gross	Decimal	->>>.>>>.>>9.99	ap_invoice.ain_amount	

Advanced

**Create the Page**

The Page should use the temp table ttStats as its query.

System | Sections | Page Summary 10 - Training Contractors QA | nigelon

Page: SYALPHAAMVST [Lint](#)

Header Tables:  
Body Table: ttStats  
Description:  
Notes:

Header Body Script Forms Fields Named Filters Browse Filters Fields Enquiry Used in

Body Program:  
Body Rows:  
Body Query: 1 FOR EACH ttStats

Query Condition:  Ignore Security:

Sort Limit: 0  
Body Title:  
Body Sum:  
Actions:  
Frozen Columns: 0  
Dialogue Width: 0  
Ajax Update:   
Matrix Columns:  
Fixed Columns:  
Auto Add:   
Single Add:   
Sequenced Update:   
Graph Formatting:  
Mandatory Columns:  
Essential Fields:

Condition:

Open Next

On our body fields, we need to specify just the fields we are displaying.

For each value, you must specify an ID (this can be any unique reference for each value)



## 10 Building COINS Desktops - Overview

Page: SYALPHAAMVST [Link](#)

Header Tables:  
Body Table: ttStats  
Description:  
Notes:


Function

Header Body Script Forms Fields Named Filters Browse Filters Fields Enquiry Used in

Field	Label	Width	Height	Function	Add	Upd	View As	Tab	Layout	Append	Hidden
ttStats.coj_type	Type	0	0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>
ttStats.dGross	Gross	0	0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>

### Create the Page

The Page should use the temp table ttStats as its query.

System | Sections | Page Summary 10 - Training Contractors QA | nigon 

Page: SYALPHAAMVST [Link](#)

Header Tables:  
Body Table: ttStats  
Description:  
Notes:

Function

Header Body Script Forms Fields Named Filters Browse Filters Fields Enquiry Used in

Body Program:  
Body Rows:  
Body Query:

```
1 FOR EACH ttstats
```

Query Condition:  Ignore Security

Sort Limit: 0  
Body Title:  
Body Sum:  
Actions:  
Frozen Columns: 0  
Dialogue Width: 0

Ajax Update:   
Matrix Columns:  
Fixed Columns:  
Auto Add:  Condition:  
Single Add:   
Sequenced Update:   
Graph Formatting: legend: {position:"bottom",axis: {x: {type:"category"}, y: {label: null, tick: {count: 5, format: d3.format(",.0f")}, rect: {fill: "white", opacity: 0.0}}}  
Mandatory Columns:  
Essential Fields:

The Gauge formatting is used in a chart page to set (via JavaScript) various graph axis and legend formatting.

Full details of the available parameters may be found at:

<http://c3js.org/reference.html>

The commands used in this example are:

legend: {position:value}

Currently values bottom, right and inset are supported.



axis: {x: {type:value}}

Currently values timeseries, category and indexed are supported.

axis: {y: {label:text,position}}

Text values will display the text as the axis label. Null will suppress any title on the specified axis

Position values (not used in our example) are:

If it's the horizontal axis:

inner-right [default], inner-center, inner-left, outer-right, outer-center, outer-left

If it's the vertical axis:

inner-top [default], inner-middle, inner-bottom, outer-top, outer-middle, outer-bottom

axis: {y: {tick:{count:value}}}

A value entered here will set the number of y axis ticks to show

axis: {y: {tick:{format:d3.format.(specifier)}}

Returns a new format function with the given string specifier. (Equivalent to `locale.numberFormat` for the default U.S. English locale.) A format function takes a number as the only argument, and returns a string representing the formatted number. The general form of a specifier is:

`[[fill]align][sign][symbol][0][width][,][.precision][type]`

The fill can be any character other than "{" or "}". The presence of a fill character is signaled by the character following it, which must be one of the align options.

The align can be:

("<") Forces the field to be left-aligned within the available space.

(">") Forces the field to be right-aligned within the available space. (This is the default).

("^") Forces the field to be centered within the available space.

The sign can be:

plus ("+") - a sign should be used for both positive and negative numbers.



minus ("-") - a sign should be used only for negative numbers. (This is the default.)

space (" ") - a leading space should be used on positive numbers, and a minus sign on negative numbers.

The symbol can be:

currency ("\$\$") - a currency symbol should be prefixed (or suffixed) per the locale.

base ("#") - for binary, octal, or hexadecimal output, prefix by "0b", "0o", or "0x", respectively.

The "0" option enables zero-padding.

The width defines the minimum field width. If not specified, then the width will be determined by the content.

The comma (",") option enables the use of a comma for a thousands separator.

The precision indicates how many digits should be displayed after the decimal point for a value formatted with types "f" and "%", or before and after the decimal point for a value formatted with types "g", "r" and "p".

The available type values are:

exponent ("e") - use `Number.toExponential`.

general ("g") - use `Number.toPrecision`.

fixed ("f") - use `Number.toFixed`.

integer ("d") - use `Number.toString`, but ignore any non-integer values.

rounded ("r") - round to precision significant digits, padding with zeroes where necessary in similar fashion to fixed ("f"). If no precision is specified, falls back to general notation.

percentage ("%") - like fixed, but multiply by 100 and suffix with "%".

rounded percentage ("p") - like rounded, but multiply by 100 and suffix with "%".

binary ("b") - outputs the number in base 2.

octal ("o") - outputs the number in base 8.



hexadecimal ("x") - outputs the number in base 16, using lower-case letters for the digits above 9.

hexadecimal ("X") - outputs the number in base 16, using upper-case letters for the digits above 9.

character ("c") - converts the integer to the corresponding unicode character before printing.

SI-prefix ("s") - like rounded, but with a unit suffixed such as "9.5M" for mega, or "1.00µ" for micro.

The type "n" is also supported as shorthand for ",g".

On our body fields, we need to specify just the fields we are plotting, and indicate which is the X value and which is the Y value.

For each value, also specify an ID (this can be any unique reference for each value)

Field	Label	Width	Height	Function	Add	Upd	View As	Tab	Layout	Append	Hidden
!Stats.co_type	Type	12	1		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>

**Layout**

Span Label: 0  
 Label Column Spans: 0 Row Spans: 0  
 Column Spans: 0  
 Mandatory:  No Break Label:  Show in Help:   
 Alignment: Default  
 Format:  
 Class:  
 Label Class:  
 Build:  
 Generate:

**Calculation**

Calculation: 1

**Scripts**

onBlur: 1

onChange: 1

onOK: 1

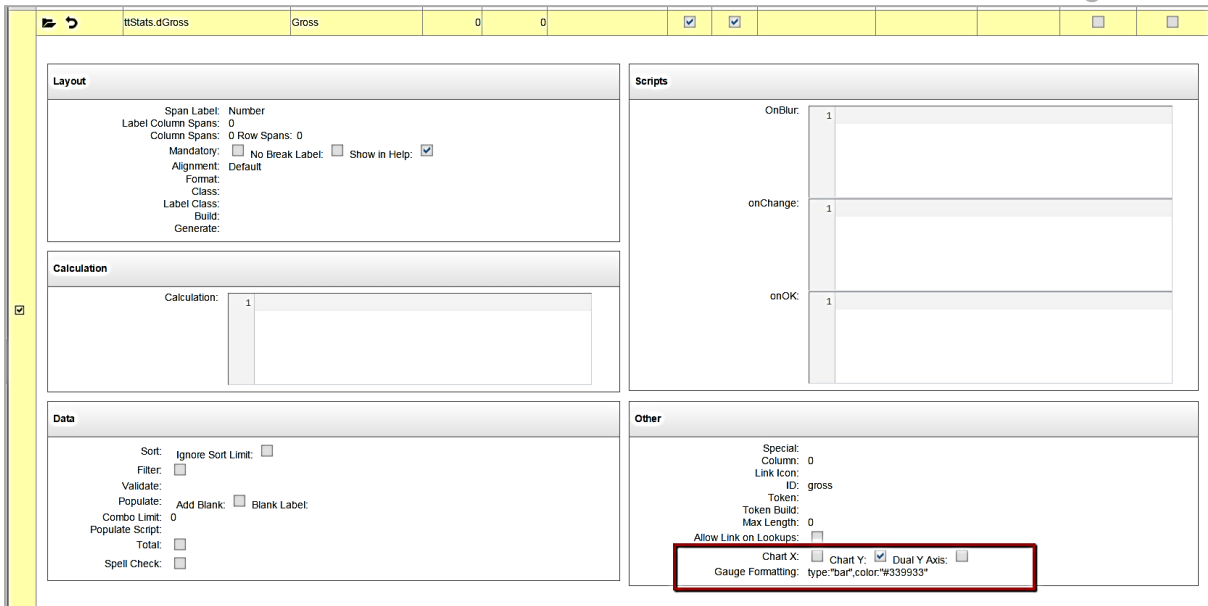
  

**Data**

Sort:  Ignore Sort Limit:   
 Filter:   
 Validate:  
 Populate: Add Blank:  Blank Label:  
 Combo Limit: 0  
 Populate Script:  
 Total:   
 Spell Check:

**Other**

Special:   
 Column: 0  
 Link Icon:  
 ID:  
 Token:  
 Token Build: 0  
 Max Length:  
 Allow Link on Lookups:   
 Chart X:  Chart Y:  Dual Y Axis:   
 Gauge Formatting:



In the Gauge Formatting field on the Y axis value, the value type:"bar",color:"#339933" sets the graph type as a Bar Chart and assigns a colour hex code to the chart. It may be good practice to assign a colour as there is a possibility of a white tile displaying a white graph – which would appear to the user as an empty chart.

### Desktop

Adding the Function to a desktop will produce the following output:

