Oracle® Banking Enterprise Limits and Collateral Management

Development WorkBench Screen Development

Release 14.8.0.0.0
G32523-01
April 2025

Oracle Banking Enterprise Limits and Collateral Management Development WorkBench Screen Development, Release 14.8.0.0.0

G32523-01

Contents

1 Preface

1.1 Purpose

This guide is designed to help acquaint you with the Oracle Banking Enterprise Limits and Collateral Management (ELCM) application. This guide provides answers to specific features and procedures that the user need to be aware of the module to function successfully.

User can further obtain information specific to a particular field by placing the cursor on the relevant field and pressing <F1> on the keyboard.

1.2 Audience

This guide is intended for the following User/User Roles:

Table 1-1 Audience

Role	Function
Back office data entry clerk	Input functions for funds
Back office managers/officers	Authorization functions
Product Managers	Product definition and authorization
End of day operators	Processing during end of day / beginning of day

1.3 Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customer access to and use of Oracle support services will be pursuant to the terms and conditions specified in their Oracle order for the applicable services.

1.4 Critical Patches

Oracle advises customers to get all their security vulnerability information from the Oracle Critical Patch Update Advisory, which is available at Critical Patches, Security Alerts and Bulletins. All critical patches should be applied in a timely manner to ensure effective security, as strongly recommended by Oracle Software Security Assurance.

1.5 Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

1.6 Basic Actions

Table 1-2 Basic Actions

Action	Description
Approve	Used to approve the initiated report. This button is displayed, once the user click **Authorize**.
Audit	Used to view the maker details, checker details, and report status.
Authorize	Used to authorize the report created. A maker of the screen is not allowed to authorize the report. Only a checker can authorize a report, created by a maker.
Close	Used to close a record. This action is available only when a record is created.
Confirm	Used to confirm the performed action.
Cancel	Used to cancel the performed action.
Compare	Used to view the comparison through the field values of old record and the current record. This button is displayed in the widget, once the user click **Authorize**.
Collapse All	Used to hide the details in the sections. This button is displayed, once the user click **Compare**.
Expand All	Used to expand and view all the details in the sections. This button is displayed, once the user click **Compare**.
New	Used to add a new record. When the user click **New**, the system displays a new record enabling to specify the required data.

Table 1-2 (Cont.) Basic Actions

Action	Description
OK	Used to confirm the details in the screen.
Save	Used to save the details entered or selected in the screen.
View	Used to view the report details in a particular modification stage. This button is displayed in the widget, once the user click **Authorize**.
View Difference only	Used to view a comparison through the field element values of old record and the current record, which has undergone changes. This button is displayed, once the user click **Compare**.
Unlock	Used to update the details of an existing record. System displays an existing record in editable mode.

1.7 Related Documents

For more information refer to the Oracle Banking manuals on:

- Development of Launch Forms and Others Screens
- Enterprise Collaterals User Guide
- Enterprise Limits and Collaterals Common User Guide

1.8 Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1.9 Screenshot Disclaimer

Personal information used in the interface or documents is dummy and does not exist in the real world. It is only for reference purposes.

1.10 Acronyms and Abbreviations

The list of the acronyms and abbreviations used in this guide are as follows:

Table 1-3 Acronyms and Abbreviations

Acronyms	Abbreviations
CIF	Customer Information File

Table 1-3 (Cont.) Acronyms and Abbreviations

Acronyms	Abbreviations
CASA	Current Account and Savings Account
DDA	System that holds the CASA account and balances
ELCM	Enterprise Limits and Collateral Management
ECA	External Credit Approval
FCUBS	Oracle FLEXCUBE Universal Banking Solution
GW	Gateway
HTTP	Hyper Text Transfer Protocol
ID	Identification Number
Mark EOTI	Mark End of Transaction Input
Mark TI	Mark Transaction Input
OFSAA	Oracle Financial Services Analytical Applications
ORMD	Oracle Revenue and Billing Management
PK	Primary Key
RDBMS	Relational Data Base Management System
SMS	Security Services
UI	User Interface
VD	Value Date
XML	Extensible Mark-up Language
XSD	XML Schema Definition
XSLT	Extensible Stylesheet Language Transformations

1.11 Symbols and Icons

The lists of symbols, buttons and shortcut key that are used in the application to perform various tasks are covered in this topic.

Table 1-4 Symbols and Icons

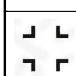
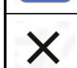
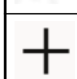
Icons	Function
	Perform search
	Minimize
	Navigate to the next record
	Navigate to the previous record
	Toggle OFF
	Toggle ON
	Delete
	Click this icon to add a new row.

Table 1-4 (Cont.) Symbols and Icons

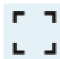
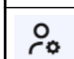
Icons	Function
	Click this icon to delete an existing row.
	List view
	Maximize
	Navigate to the first record
	Navigate to the last record
	Advance search
	Search record
	Save the record
	Reset the record
	Clear the record

Table 1-5 Symbols and Icons - Audit Details

Icons	Function
	A user
	Branch details
	Date and Time

1.12 Prerequisite

Specify the **User ID** and **Password**, and login to **Home** screen.

2 Overview of Screen Development for Oracle FLEXCUBE Universal Banking

This topic provides an overview of screen development for Oracle FLEXCUBE Universal banking.

Oracle FLEXCUBE ODT provides the developer with a user-friendly console for designing and developing screens for Oracle FLEXCUBE Universal Banking.

ODT assists developers in designing screens with the capability of generating front-end scripting files, PL/SQL Packages, Static data scripts, XSDs, Excel templates, and HTML files. This generated code performs validations and does some processing which is common across screens in FLEXCUBE; only the Business logic specific to the screen has to be added by the Developer in the back end and front end units.

Example 2-1

Release Name: FC 11.3

Release Type: KERNEL, CLUSTER, CUSTOM

- Radxml
  This topic provides an overview of the Radxml file.
- Extensible Development
  This topic provides an overview of the extensible development of FLEXCUBE.
- Design Steps
  This topic explains the systematic instructions to developing a screen in ODT.

2.1 Radxml

This topic provides an overview of the Radxml file.

ODT saves all the activities carried out by the developer in an XML file hereby referred to as Radxml. The persistence of the screens is achieved through Radxml. All the units required for the working of a screen can be generated from its Radxml.

If some changes are required on the screen in a future release, the same Radxml can be loaded and changes can be done on this Radxml. ODT can segregate the changes done on different releases and save the Radxml accordingly.

Radxml will adhere to the naming convention: **Function Id name + _RAD.xml**

For Example: **FTDTRONL_RAD.xml**

2.2 Extensible Development

This topic provides an overview of the extensible development of FLEXCUBE.

Table 2-1 Extensible Development Stages

Extensible Development Stages	Description
KERNEL	This refers to the core product. For Example: FC 11.3
CLUSTER	This refers to any region-specific enhancements/ developments done on top of the **KERNEL** product. The changes done in a **CLUSTER** pack will be noninvasive to the **KERNEL** product. For Example: FC 11.3 India cluster
CUSTOM	Any enhancements required by the customer/bank are done on this level. It will be noninvasive of the changes done in **KERNEL** and **CLUSTER** packs. For Example: Customizations for the MODEL bank on FC 11.3 India cluster

ODT can segregate the changes done on different releases and save the Radxml accordingly. It generates the files depending on the release type and developers can add the business logic to the release type-specific files. Thus the code remains noninvasive of the existing code.

2.3 Design Steps

This topic explains the systematic instructions to developing a screen in ODT.

1. Identifying the data sources and their relations.
2. Logically grouping the data sources into Data Blocks.
3. Designing Screen Layout
4. Logically grouping the Block Fields into Field sets
5. Attaching Call forms and launch forms if any
6. Defining LOVs
7. Designing Summary
8. Defining Actions

   Refer respective sections for detailed explanation of each step.

   To save Radxml, while Development save radxml at constant intervals. Click on **save** icon in the top right for having the work. Radxml would be saved in the user directory maintained

3 Header Information

This topic provides systematic instructions to update header information.

1. Login to the Oracle FLEXCUBE Universal Banking ODT using the credentials maintained.

   Refer Development Workbench – AdministrationDevelopment Workbench – Administration for creating users.

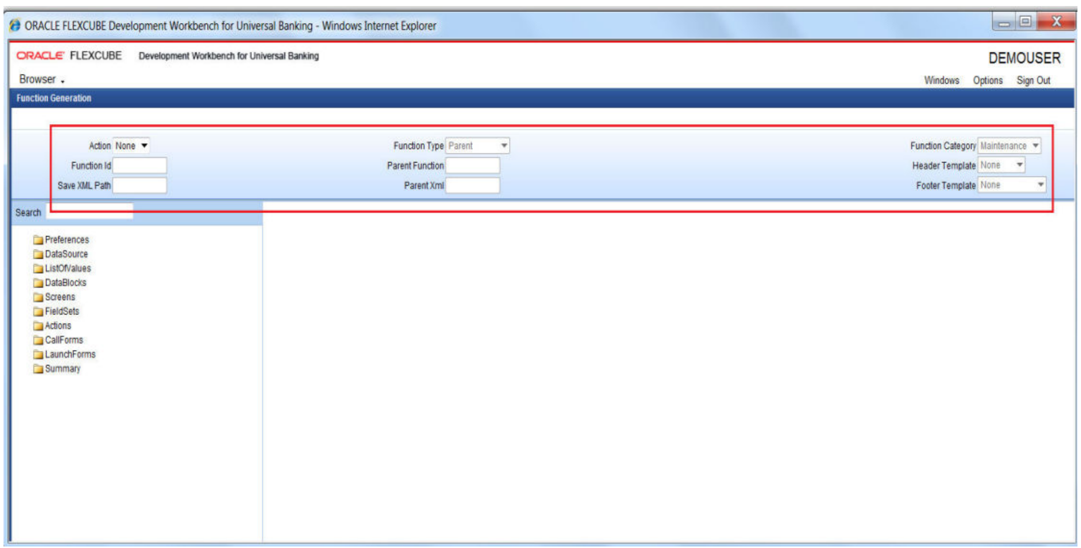
2. Map the session to the release and environment as required

   Refer Development Workbench – Getting Started for detailed explanation.

3. Click on Function Generation node in the browser tree found in the Landing page of ODT.

   Function Generation screen displays.

   Figure 3-1 Function Generation

4. On **Function Generation** screen, specify the following.

   Table 3-1 Function Generation - field Description

Field	Description
Action	New and Load options are provided for this field. For a new screen development, select the action as New; if an existing screen radxml has to be loaded for customization select Load option. If the action is load then corresponding radxml has to be loaded using browser option in Save Xml Path; all the header information will get populated

Table 3-1 (Cont.) Function Generation - field Description

Field	Description
Function Id	If the Action is selected as New, the function Id name needs to be specified. Function Id is the unique name with which a screen is identified. Function Id name should follow the FLEXCUBE standard naming convention. • Function Id name to have maximum length of 8 characters • For detail screens the third character should be **D** • For report screens the third character should be **R** • For call form function ids the third character should be **C** • First 2 characters should specify the module name for which the particular function id is used.
Function Type	Function Type can be Parent or Child or Screen Child (based on the screen which has to be designed). For more information on the #unique_29/unique_29_Connect_42_TABLE_BLS_JSM_4WBtable.
Parent Function	This field is applicable only if the function type is child/screen child and this field will be populated when the parent RAD xml is loaded. This is a read only field.
Save Xml Path	The label description of the field will change depending on the action .If the action is load, ODT attaches a Browse button to it so that user can browse the radxml and load it. If the action is New, save xml path is optional. If provided, then the generated units will be saved in the path mentioned. Note that the value in the Save Xml Path will be used only if the Save Format is Client Path and if the User has given “CURRENT_DIRECTORY” in the User Preferences Work Directory.
Parent Xml	This field is also applicable only if the Function type is child/screen child. If the Function Type is child/screen child user has to load the radxml of the Parent Function using browse button provided to this field. It is non editable field if the action is **Load**.
Function Category	Provide Function category depending on the type of screen being developed. ODT provides the following options: For more information on the filed, refer to the #unique_29/unique_29_Connect_42_TABLE_MYD_3QM_4WB table.
Header Template	A template can be selected for header. The following options are provided.
Footer Template	A template can be selected for footer. The following options are provided. For more information on the field, refer to the #unique_29/unique_29_Connect_42_TABLE_KHS_XTM_4WB table.

Table 3-2 Function Type

Field	Description
Parent	This is the default option and can be used for normal screen development.
Child	This option can be selected if the screen has to be the child of another screen .i.e. inherits all the properties of another screen which will be its parent. Properties can be modified in the child level.

Table 3-2 (Cont.) Function Type

Field	Description
Screen Child	This option can be selected if the screen has to be the screen child of another screen, i.e. it inherits all properties from its parent. Only screen layout changes can be done in the screen child screen.

Table 3-3 Function Category

Field	Description
Maintenance	These screens are typically used to maintain static data used across the system. These screens include product definition function as well. Example: Branch Parameters Maintenance
Transaction	These screens are typically used to capture contract related data. Any operations related to contracts are performed in these screens Example: Funds Transfer Contract Input screen
Report	These screens are used to capture data required to generate a BI Publisher canned reports. Example: General Ledger balance report
Summary	If only query operation is required for the particular function Id, then function category can be selected as summary.
Others	If developer feels that existing handles provided in maintenance/ transaction screens in extensible framework is inadequate (or not necessary) for the screen; screen can be designed as others. Note that all business logic would have to be manually written by the developer for others screen.

Table 3-4 Header Template

Field	Description
None	This is the default header and should be used for all screens except workflow screens.
Process	This template can be selected for workflow screens. The following Fields will be added to the header section as part of this template. a. Workflow Reference b. Priority

Table 3-5 Footer Template

Field	Description
None	This is the default value.

Table 3-5 (Cont.) Footer Template

Field	Description
Maint audit	This template can be used for maintenance screens. Ensure that master data source has the standard audit columns. Audit block field names are reserved filed names and hence cannot be used as the name of any other block field. For more information on the field, refer to the #unique_29/unique_29_Connect_42_TABLE_N53_NVM_4WB table. **Note:** When template is selected, ODT automatically adds the fields to Data Source and Blocks and adding these fields manually to data blocks results in erroneous behavior.
Maint Process	In this template along with maintenance audit fields System would automatically add a control block and Process related fields. This template can be used for workflow maintenance screens.
Process	Only process related Fields will be added to the Footer. This can be used for work flow transaction screens. As part of process template; previous remarks, remarks, outcome and audit block will be created in the footer.

Table 3-6 Audit column and corresponding block fields

Column Name	Block Field Name
MAKER_ID	MAKER
MAKER_DT_STAMP	MAKERSTAMP
CHECKER_ID	CHECKER
CHECKER_DT_STAMP	CHECKERSTAMP
MOD_NO	MODNO
RECORD_STAT	RECSTAT
ONCE_AUTH	ONCEAUTH
ONCE_AUTH	AUTHSTAT

For example, For Funds Transfer Contract Input Screen name can be given as FTDTRONL Here FT is the module (Funds Transfer), third letter D denotes it is a normal detail screen, and Length of the function Id is 8.

If the action selected is Load, function Id field will be disabled. It will be picked up from the radxml which is loaded.

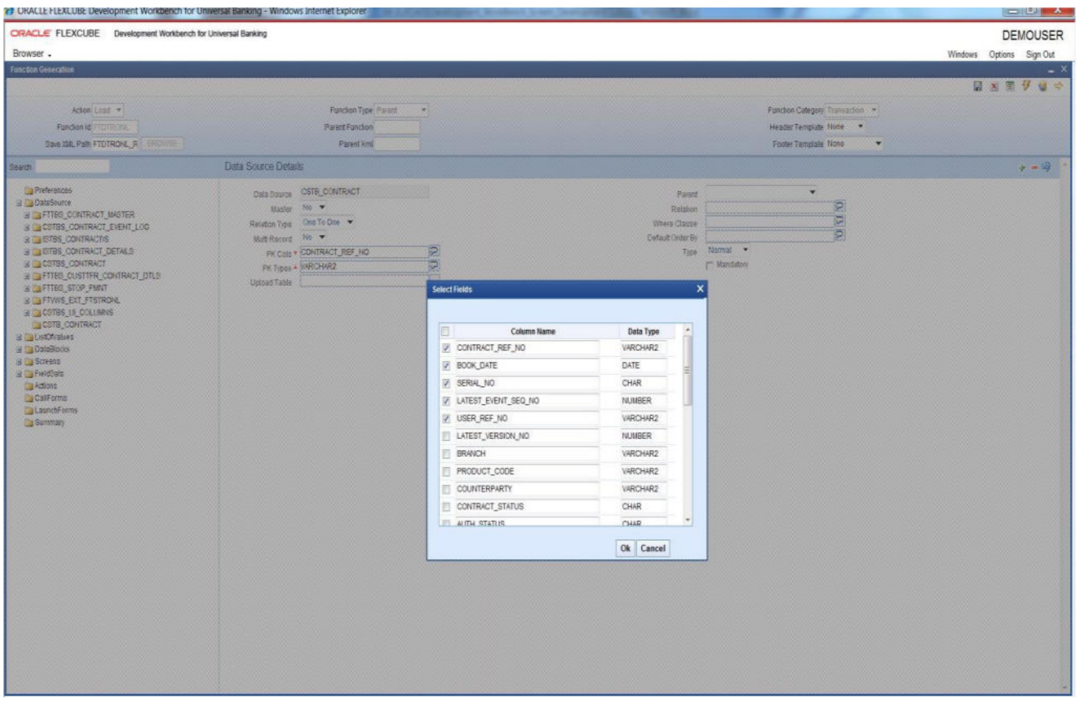
4
Add Data source columns

This table provides systematic instructions to add Data source columns.

1. Right-click on the data source to add the data source column.

   Select Fields window displays.

   Figure 4-1 Adding Fields to created Data Source

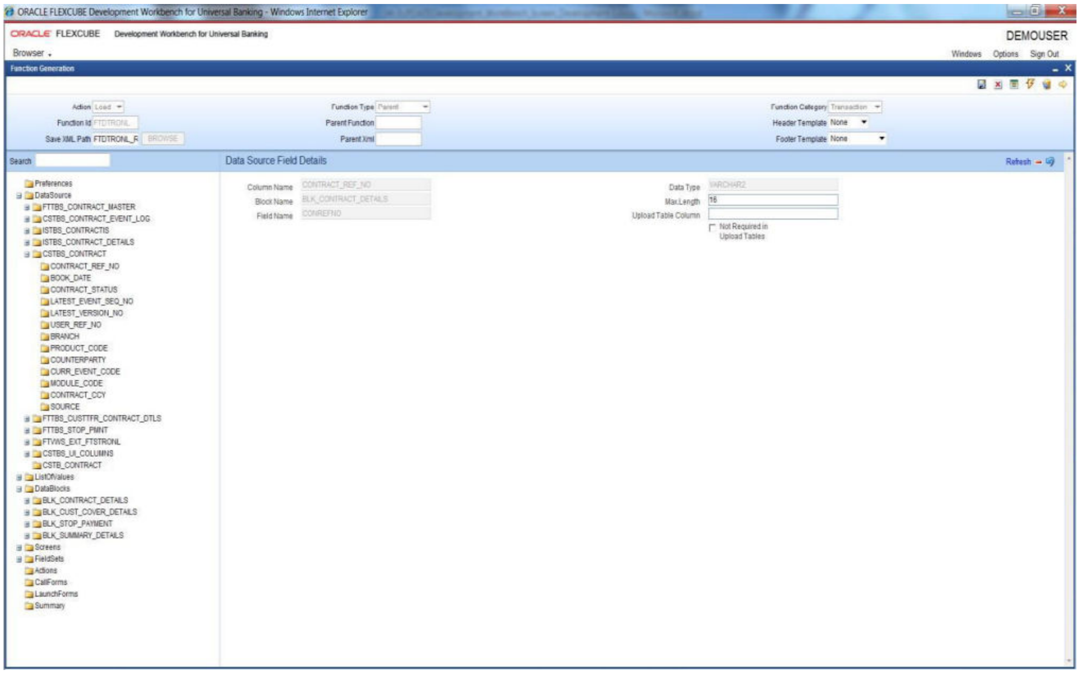
2. Select all the columns which have to be included as per the design and then click on the **OK** button.

   > **Note:**
   >
   > All the columns mentioned as primary key columns in the data source level have to be mandatorily included.

   If the footer template is **Maintenance Audit**, audit columns should not be selected for the master data source. ODT will attach the audit columns automatically while generating units.

   Data Source Field Details screen displays.

Figure 4-2 Data Source Fields Details



3. Specify the Data Source Fields details.

Table 4-1 Data Source Fields Details- Field Description

Field	Description
Column Name	This is an information column. The tool will not allow the user to change the value.
Max Length	The max length of the column will have defaulted while adding a column and it can be overwritten. This value will be considered for the field max length of the designed screen. It means, at the run time system will not allow the user to enter the text more than this length.
Data Type	This shows the data type of the column selected and the value will be populated while adding the column, this is a non-editable field.
Block Name	It is an information field, if the field is added to a block, that block name will be populated here.
Field Name	It is an information field. This displays the data block field to which the column is mapped.
Upload Table Column	If an adapter upload package is required, the column name in the upload table to which this column is mapped has to be provided. If the field is left blank, then the column name in the upload table will be assumed to be the same as the data source column name.
Not Required in Upload Tables	If the particular column is not mapped to any upload table column, then this option has to be checked.

5

Create new function in ODT

This topic provides systematic instructions to create a new function in ODT.

1. Log in to the Oracle FLEXCUBE ODT using the credentials maintained.

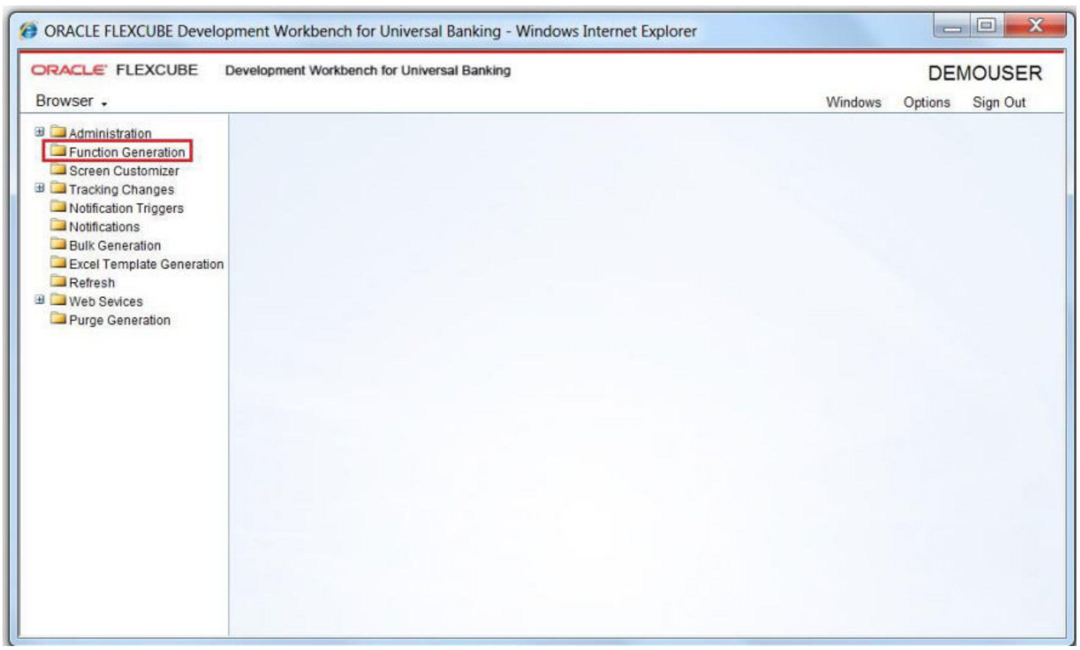
   Refer to Development Workbench – Administration document for creating the user.

2. Map the session to the release and environment as required.

   Refer to Development Workbench – Getting Started for detailed explanation.

   Development Workbench Universal Banking- Home screen displays.

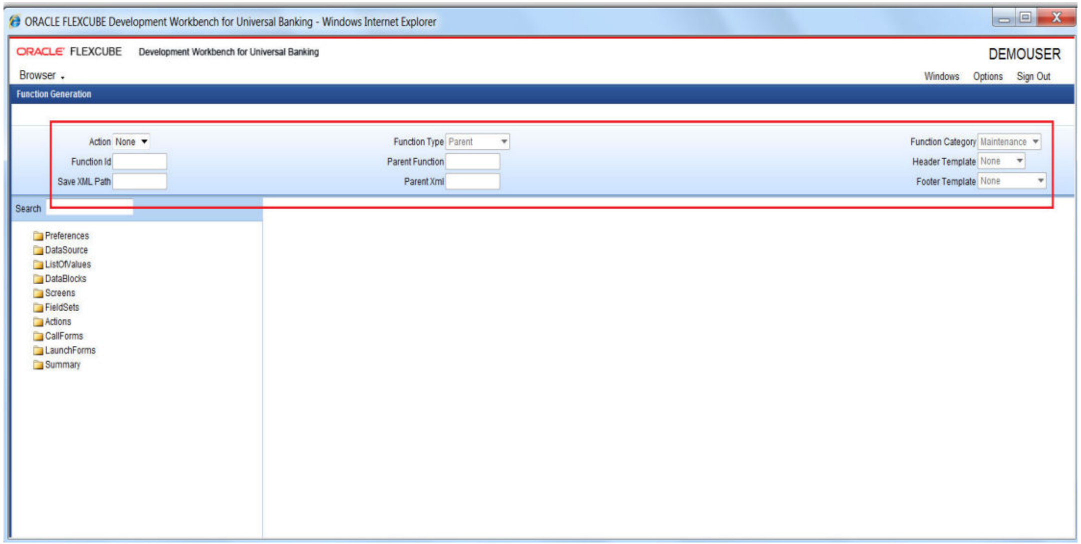
   Figure 5-1 Development Workbench Universal Banking- Home

3. Click on the **Function Generation** node in the **Browser** tree.

   Function Generation screen displays.

Figure 5-2   Function Generation



4. To create a new function in ODT, specify the details in the Header section:

Table 5-1   Header section

Field	Description
Action	New and Load options are provided for this field. For a new screen development, select the action as **New**. If an existing screen Radxml has to be loaded for customization, select the **Load** option. If the action is **Load** then the corresponding Radxml has to be loaded using **Browser** option in Save Xml Path; all the header information will get populated.
Function Id	If the Action is selected as **New**, the function Id name needs to be specified. Function Id is the unique name with which a screen is identified. Function Id name should follow the FLEXCUBE standard naming convention. • Function Id name to have a maximum length of 8 characters. • For detail screens the third character should be **D**. • For report screens the third character should be **R**. • For call form function ids the third character should be **C**. • First 2 characters should specify the module name for which the particular function id is used(recommended). For Example: For Funds, Transfer Contract Input Screen name can be given as **FTDTRONL**. Here FT is the module (Funds Transfer), third letter D denotes it is a normal detail screen, and The length of the function Id is 8. If the action selected is **Load**, the function Id field will be disabled. It will be picked up from the Radxml which is loaded.

Table 5-1 (Cont.) Header section

Field	Description
Save XML Path	The label description of the field will change depending on the action. If the action is **Load**, ODT attaches a **Browse** button to it so that the user can browse the Radxml and load it. If the action is New, save xml path is optional. If provided, then the generated units will be saved in the path mentioned. Note that the value in the Save Xml Path will be used only if the Save Format is Client Path and if the User has given **CURRENT_DIRECTORY** in the **Work Directory** under **User Preferences**.
Function Type	Function Type can be **Parent** or **Child** or **Screen Child** (based on the screen which has to be designed). • **Parent**: This is the default option and can be used for normal screen development. • **Child**: This option can be selected if the screen has to be the child of another screen; that is inherits all the properties of another screen which will be its parent. Properties can be modified at the child level. • **Screen Child**: This option can be selected if the screen has to be the screen child of another screen, that is it inherits all properties from its parent. Only screen layout changes can be done on the screen child screen. Refer respective documents for detailed explanation on **Parent**/**Child**/**Screen Child** screens.
Parent Function	This field is applicable only if the function type is child/screen child and this field will be populated when the parent Radxml is loaded. This is a read-only field.
Parent XML	This field is also applicable only if the Function type is child/screen child. If the Function Type is child/screen child user has to load the Radxml of the Parent Function using the browse button provided to this field. It is a non-editable field if the action is **Load**.

Table 5-1 (Cont.) Header section

Field	Description
Function Category	Provide Function category depending on the type of screen being developed. ODT provides the following options: • **Maintenance**: These screens are typically used to maintain static data used across the system. These screens include product definition functions as well. For Example: Branch Parameters Maintenance • **Transaction**: These screens are typically used to capture contract-related data. Any operations related to contracts are performed on these screens. For Example: Funds Transfer Contract Input screen • **Report**: These screens are used to capture data required to generate BI Publisher canned reports. For Example: General Ledger balance report • **Summary**: If only query operation is required for the particular function Id, then the function category can be selected as Summary. • **Others**: If the developer feels that existing handles provided in maintenance/ transaction screens in the extensible framework are inadequate (or not necessary) for the screen; the screen can be designed as others. Note that all business logic would have to be manually written by the developer for other screens.
Header Template	A template can be selected for the header. The following options are provided. • **None**: This is the default header and should be used for all screens except workflow screens. • **Process**: This template can be selected for workflow screens. The following Fields will be added to the header section as part of this template. a. Workflow Reference b. Priority

Table 5-1 (Cont.) Header section

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>Footer Template</td><td>A template can be selected for the footer. The following options are provided.
None: This is the default value
Maintenance Audit: This template can be used for maintenance screens. Ensure that the master data source has the standard audit columns.

Table 5-2 Maintenance Audit column
<table><tr><th>COLUMN NAME</th><th>BLOCK FIELD NAME</th></tr><tr><td>MAKER_ID</td><td>MAKER</td></tr><tr><td>MAKER_DT_STAMP</td><td>MAKERSTAMP</td></tr><tr><td>CHECKER_ID</td><td>CHECKER</td></tr><tr><td>CHECKER_DT_STAMP</td><td>CHECKERSTAMP</td></tr><tr><td>MOD_NO</td><td>MODNO</td></tr><tr><td>RECORD_STAT</td><td>RECSTAT</td></tr><tr><td>ONCE_AUTH</td><td>ONCEAUTH</td></tr><tr><td>AUTH_STAT</td><td>AUTHSTAT</td></tr></table>
Audit block field names are reserved filed names and hence cannot be used as the name of any other block field.

Note:
when the template is selected, ODT automatically adds the fields to Data Source and Blocks, and adding these fields manually to data blocks results in erroneous behavior.
Maintenance Process: In this template along with maintenance audit fields system would automatically add a control block and Process related fields. This template can be used for workflow maintenance screens.
Process: Only process-related Fields will be added to the Footer. This can be used for workflow transaction screens. As part of the process template; previous remarks, remarks, outcomes, and audit block will be created in the footer.</td></tr>
</table>

6

Create New Data Source

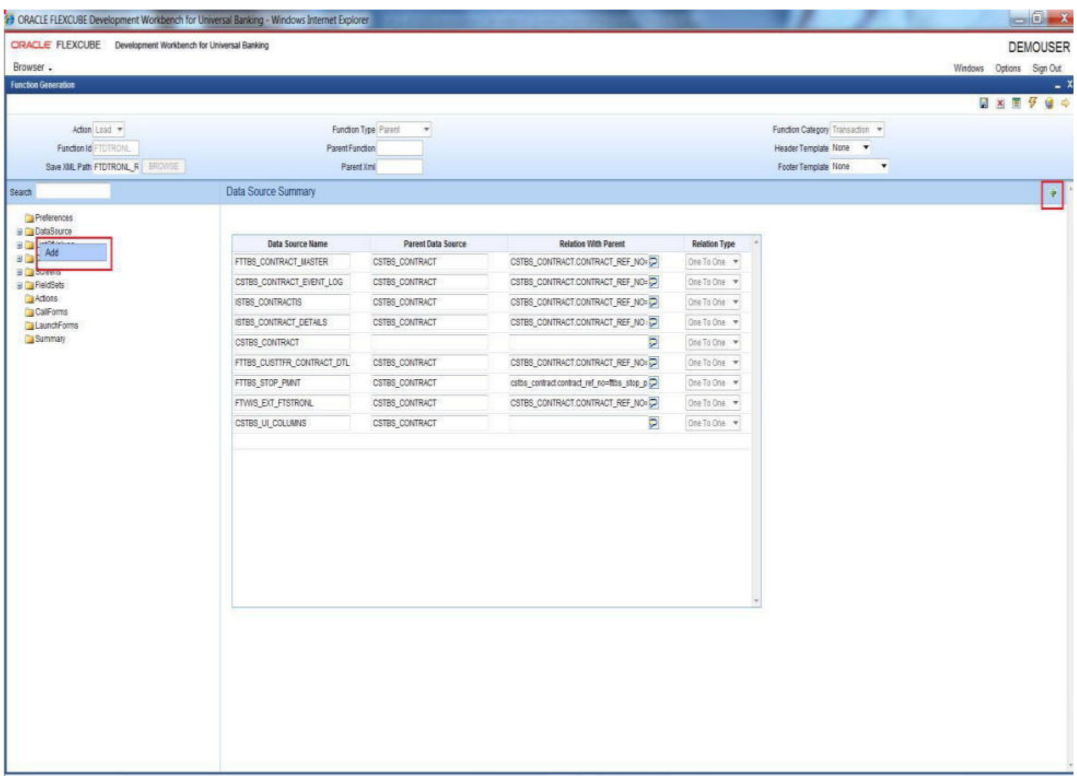
This topic provides systematic instructions to create a new Data Source.

The first step in developing Functions is to identify the tables and views involved in the Function being developed. Relations among these data sources and the types of data sources need to be identified based on the functionality.

1. Under **Function Generation** screen, right-click on the **DataSource** node and select the **Add** option to create a new Data Source.
2. Or Left click on the **DataSource** node.

   Data Source Summary screen displays.

Figure 6-1 Data Source Summary



3. Click **Add DataSource (+)** icon on the top right of the screen to create a new Data Source.
4. In the **Data Source Summary** screen, select the **Data Source Name** from the list of values provided.

   A list of values will fetch all objects from the business schema.

Figure 6-2 Add Table window

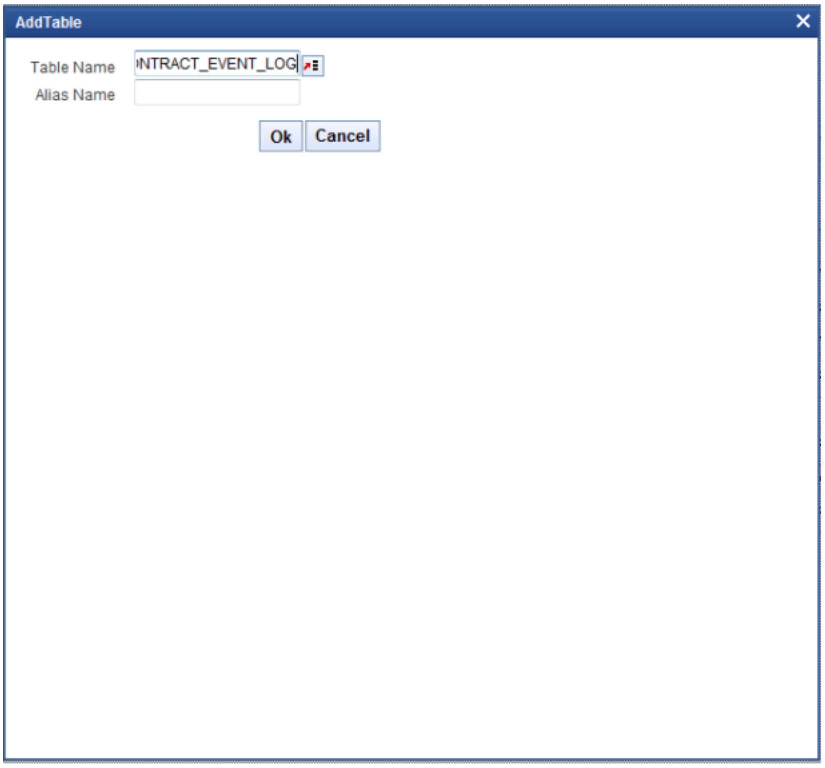
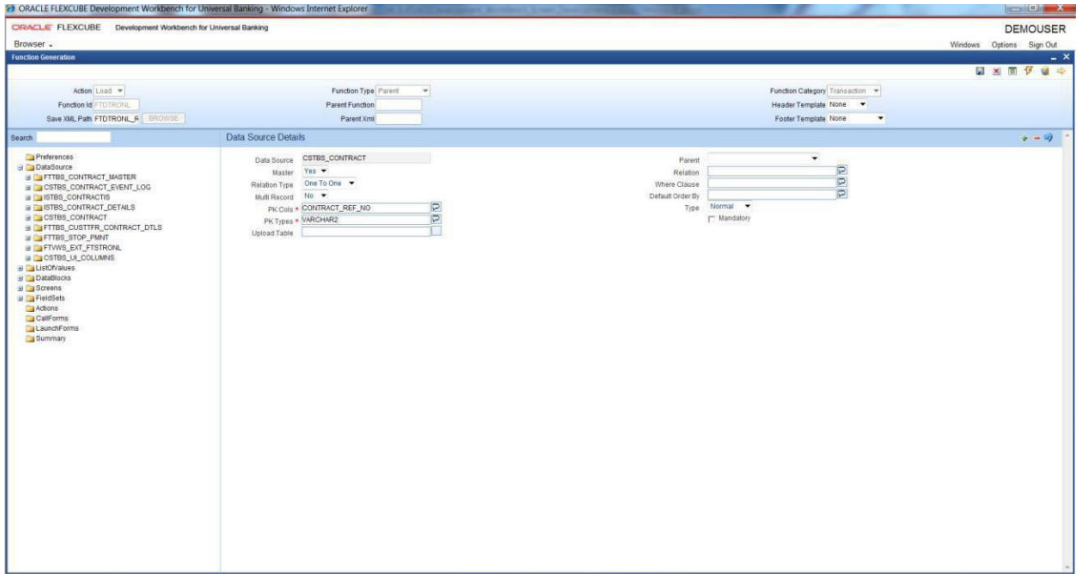


Table 6-1 AddTable- Field Description

Field	Description
Alias Name	If the same table is used more than once in the design, the developer can differentiate the same using different alias names.

Data source name will have table name appended with the alias name separated by __. For Example: If the table name is **CSTB_UI_COLUMNS** and alias name is **A**. The data source name will be **CSTB_UI_COLUMNS__A**

Figure 6-3 Data Source Properties



5. Specify properties of the created data source.

Table 6-2 Data Source Properties

Field	Description
Data Source	This is a read-only field. Data source name will have defaulted to this field.
Master	This file identifies whether this data source is the master data source for the screen. Every screen should have one (and only one) master data source.
Relation Type	Relation type can be selected for the selected data source with the parent, **One To One** or **One To Many**. The relation should be **One To Many** if the parent data source is a Single Record entry and the child data source is Multi Record.
Multi Record	This field tells about the type of data source in the screen, whether it is a multiple Entry record or the single Entry Record data source, Select the field value **Yes** or **No** accordingly. Master Data source cannot be of multi-record type.
Pk Cols & Pk Type	**Pk Cols & Pk Type** fields are mandatory. Provide the Primary key columns of the data source separated by tilde (~) in the **PK Col** field and its corresponding data types in the PK Type field(also separated by tilde). For Example: If Pk Cols are CONTRACT_REF_NO (VARCHAR2) and VERSION_NO(NUMBER), Pk Col can be provided as CONTRACT_REF_NO~VERSION_NO and Pk Type can be provided as VARCHAR2~NUMBER

Table 6-2 (Cont.) Data Source Properties

Field	Description
Parent	The parent data source has to be mentioned for all data sources except the master data source. Select List provides all the data sources created till that point. The developer can choose the data source from the list. For master data source, the parent should not be provided.
Relation	Relation with a parent has to be specified for all data sources except the master data source. Keep the parent data source on the left side of the relation. For Example: If the parent is STTM_CUSTOMER and the child is STTM_CUST_ACCOUNT, and the relationship is based on CUSTOMER_NO and BRANCH_CODE, it has to be provided as STTM_CUSTOMER.CUSTOMER_NO= STTM_CUST_ACCOUNT.CUSTOMER_NO AND STTM_CUSTOMER.BRANCH_CODE= STTM_CUST_ACCOUNT.BRANCH_CODE **Note:** The relationship can be only with parent maintained and the current data source; a third table cannot be introduced in the relationship. Only simple Relational operators and **equal to** can be used in the relationship.
Where Clause	This is an optional field. If only selected records of a particular data source are required, then where clause can be provided. Keyword **WHERE** need not be provided. For Example: Where clause can be given as BRANCH_CODE=GLOBAL.CURRENT_BRANCH, The values of **Where Clause** field will be used in generating the query statements for the current data source. During the query of the record, this clause will be added in **fn_sys_query** of the generated main package.
Default Order By	This is an optional field. This can be provided for multi-record data sources. While querying they would be fetched in the order by clause provided. Keyword **ORDER BY** is not required. For Example: If the user wants to order by EVNET_SEQ_NO, the default order can be provided as EVENT_SEQ_NO.

Table 6-2 (Cont.) Data Source Properties

Field	Description
Type	The data source can be selected depending on the Type requirement of the screen design. Options provided are: • **Normal**: Data from the screen will be persisted in the table. Code for persistence would be available in the generated package. • **Query**: The data source can be used for only querying the data; ODT will generate packages without insert or update statements on this data source. • **InOnly**: Tool will generate the packages without insert and update on this data. Request XML will contain the data source while the response won't contain it. • **Summary**: This data source can be used for summary and this data source will not be considered while generating the packages.
Mandatory	If at least one record has to be provided for this data source, then the data source can be provided as mandatory.
Upload Table	If adapter upload is required for the function Id, then the data source should be mapped to its corresponding upload table.

7
Set up Preferences

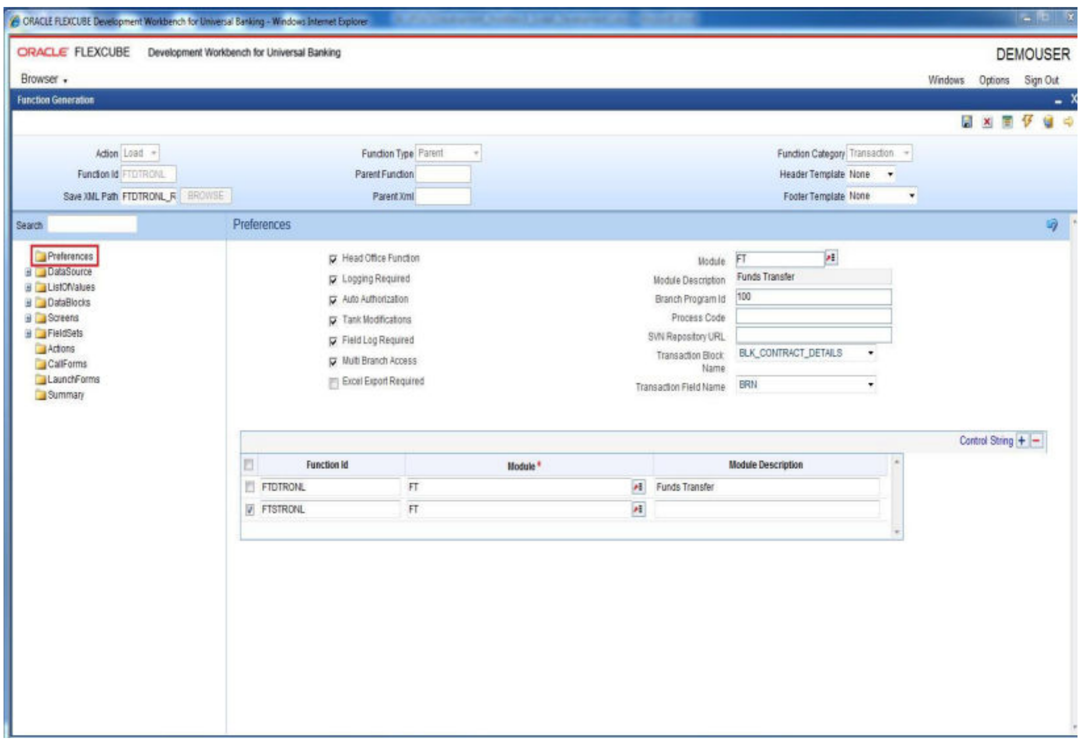
This topic provides systematic instructions to set up preferences for the new function in ODT.

1. Under the **Function Generation** screen, click on the **Preferences** node to set the Function ID level preferences.

   Function ID level preferences like **Module**, **Logging required**, **Tank Modifications** Main menu, **Sub-Menu1**, **Sub-Menu2** are maintained through the **Preferences** screen in ODT.

   Preferences screen displays.

   Figure 7-1 Preferences Screen in ODT Function Generation

   The data maintained in **Preferences** Screen will be used for generating static data script for tables **SMTB_MENU**, **SMTB_FUNCTION_DESCRIPTION**, **SMTB_ROLE_DETAIL** and **SMTB_FCC_FCJ_MAPPING**.

2. Specify the details in the **Preferences** screen.

Table 7-1 Preferences

Field	Description
Module	It captures the Module of the function ID. The developer can choose the module name from the list of values provided. The list of values is populated based on the modules maintained in the **SMTB_MODULE** table of the business schema (current FLEXCUBE environment to which tool is mapped). Module Code has to be provided mandatorily. Module name provided would be reflected in the script for **SMTB_MENU** generated by ODT (in module column). This will also be considered while naming the packages generated by the tool. The first two characters of the packages generated will be taken from the module code maintained.
Module Description	Module description gets defaulted based on the module code selected.
Head Office Function	It captures whether the function is a head office function or not. It will be reflected in the static script for **SMTB_MENU** in column **HO_FUNCTION**. If the function is a head office function, only query operation will be possible at all the other branches for the particular screen.
Logging Required	It captures whether logging is required for the function or not. This will also be reflected in the script generated for **SMTB_MENU** (column **LOGGING_REQD**). If this option is selected, all the requests and responses for the function Id will be logged in **CSTB_MSG_LOG**. This is used for the **View Change** option.
Auto Authorization	It captures whether Auto Authorization is allowed for the function or not. This will also be reflected in the script generated for **SMTB_MENU** (column **AUTO_AUTH**). Note that auto authorization is possible only if it is allowed at function id level, user-level, and branch level.
Tank Modifications	It captures whether a modification is to be tanked for the function or not. This will be reflected in the **TANK_MODIFICATIONS** column of the **SMTB_MENU** table script. If tank modification is enabled, then the record in that screen would be logged to logging tables and taken up for processing (untanking) in the later stage during EOD operations. Currently, this is applicable only for maintenance screens.
Field Log Required	It captures whether field Logging is required for the function or not. This will be reflected in the **FIELD_LOG_REQD** column in **SMTB_MENU**. If the field log required is enabled, then all operations on the screen will be logged to logging tables (**STTB_FIELD_LOG**, **STTB_RECORD_LOG**, etc). Currently, this is applicable only for maintenance screens.
Excel Export Required	This field captures whether the option to export records from summary screen to excel is required.
Multi Branch Access	It captures whether multi-branch access is required for the function Id or not. This will be reflected in the **MULTIBRANCH_ACCESS** column of the **SMTB_MENU** table script generated. If multi-branch access is allowed, then records of different branches for the screen can be modified from a single branch.
Transaction Block Name	It captures the transaction Block Name. This is applicable only if multi-branch access is allowed for the screen. Select the block from the select list which contains the field for branch code.

Table 7-1 (Cont.) Preferences

Field	Description
Transaction Field Name	It captures the transaction Field Name. This is applicable only if multi-branch access is allowed for the screen. The select list provides all the block fields for the transaction block selected. Choose the field for branch code from the list. **Transaction Block Name** and **Transaction Field Name** will be reflected in the system JavaScript file (SYS js) generated by ODT. The developer has to code for querying the records based on the branch code value of this field.
Branch Program ID	It captures the branch program ID for the function.
Process Code	This can be used to map which process needs to be initiated during screen launch. This is used for workflow screens.
SVN Repository URL	This is applicable only if integration to SVN (version control tool) is required. Path of the SVN repository till the module needs to be provided in this field.
Control String	Control String defines the operations which can be done on a particular screen. Control String has to be maintained for function ID. Select the function ID for which control string has to be modified and click on Control String. All the available operations can be found. Check all the operations which have to be allowed for the particular function ID and click **OK**. Note that for summary screens, the control string will be disabled. Normally control screens need to be provided only for the detailed screen. **REVERSE**, **ROLLOVER**, **CONFIRM**, **LIQUIDATE**, **HOLD** operations are applicable only for transaction screens. Control String will get reflected in the **CONTROL_STRING** column in **SMTB_MENU** for the particular function id. It will also be reflected in the script for **SMTB_ROLE_DETAIL**.
Function ID, **Module** and **Module Description**	In the multiple-entry block, the developer needs to maintain all the related function ID names for the screen. ODT will default the name of the function ID to the first row of the multiple entries along with the module maintained earlier. If any other function id is required for the particular screen, the developer has to add the row. For Example: For **FTDTRONL** screen, the developer designs a detail screen. The developer also wants to add one summary screen to the screen as well as a gateway function ID for web services. For this, the developer can add two new columns **FTSTRONL** (for a summary) and **FTGTRONL** (for gateway operations) to the block. Each row in this block will be reflected as one row in **SMTB_MENU**, **SMTB_FUNCTION_DESCRIPTION**, **SMTB_ROLE_DETAIL** and **SMTB_FCC_FCJ_MAPPING**.

8

Data Sources- Guidelines and Best Practices

This topic provides a list of guidelines and best practices for creating data sources.

Below are the basic steps to be followed while creating data sources:

1. Identify the tables and views involved in the Function being developed.
2. Establish the hierarchy, Relation, and Relation Types among these Data Sources.
3. Ensure that the correct and full relation is mentioned in the Relation.
4. Identify the type of Data Source. It could be Normal, Summary, In Only, or Query Source.
5. Ensure that there is only one master Data source for the Function.
6. Based on the number of records that the data source might have for the function Id, set Multi Record to Y/N.
7. System automatically defaults the PK Column information from STTB_PK_COLS while adding the data source. Based on the specific requirement of the function ID this can be modified.
8. If the data source is mandatory, that is if it is a multi-record data source and has to have at least one record or if it's a single record data source and is mandatory to have the record, check the mandatory flag.
9. In case there is a need to have a default where clause or order by clause, mention the same in the respective fields.

Best Practices

Below are the practices to be followed while creating data sources:

- Table/View/Synonym Names should adhere to the standard FLEXCUBE naming conventions. Tables or views should have the 5th character as _ **(underscore)**. The name should not have underscores together (example: ACTB_TEST__ODT is wrong). Synonyms should have an **S** appended before the first underscore of their table/view name. For Example: Synonym for ACTB_DAILY_LOG should be ACTBS_DAILY_LOG
- Avoid using views as much as possible. Don't create a view data source with type as NORMAL. This will result in insert statements on the view in the packages generated.
- Views can be used for query-only purposes, i.e. select the data source type as the query for views. These can be used for designing summary data sources or a query block.
- PK Cols and PK types need not be the same as the primary key of the tables. It depends on the design logic.
- If the data source is designed with relation type as 1:N with its parent, then it should have at least one more Pk col than its parent (assuming the relationship is based on pk cols). For Example: Assume STTM_CUSTOMER is the parent data source (1:1) with Pk col as CUSTOMER_NO; STTM_CUST_ACCOUNT is the child of STTM_CUSTOMER with 1: N relation with the parent. Here STTM_CUST_ACCOUNT should have at least 2 Pk Cols so that each record of the multi-record can be uniquely distinguished. CUSTOMER_NO and ACCOUNT_NO can be provided as Pk col so that each record can be identified with its account no. Relationship can be STTM_CUSTOMER.CUSTOMER_NO= STTM_CUST_ACCOUNT.CUSTOMER_NO

- The parent data source has to be above all of its child data sources in the tree.

9

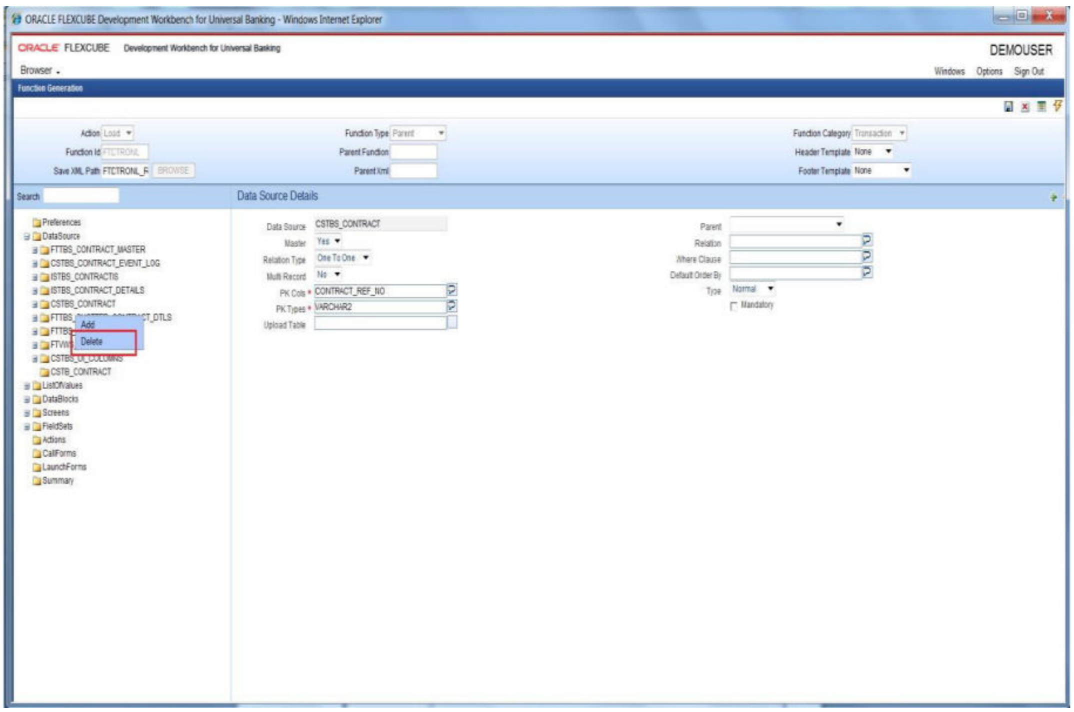
Delete Data Sources

This topic provides systematic instructions to delete data sources.

1. Under **DataSource** node, right-click on the data source to be deleted and select the **Delete** option.
2. Or click on the **Delete** icon in the top left of the data source screen to delete the data sources.

   When a data source is deleted, all the references to that data source or its columns will also be deleted from the Radxml, that is Data block fields referring to the particular data source column and all the references to those data block fields.

Figure 9-1 Deleting a Data Source



> **Note:**
>
> The deletion of any element from Radxml is allowed only if that element has been created in the current release. If the data source has been created in a previous release, then the developer won't be allowed to delete the data source.

Similarly, the data source column alone can be deleted from the data source subjected to the condition mentioned above. Proper care has to be done while designing data sources as redesigning would be impossible in a later release.

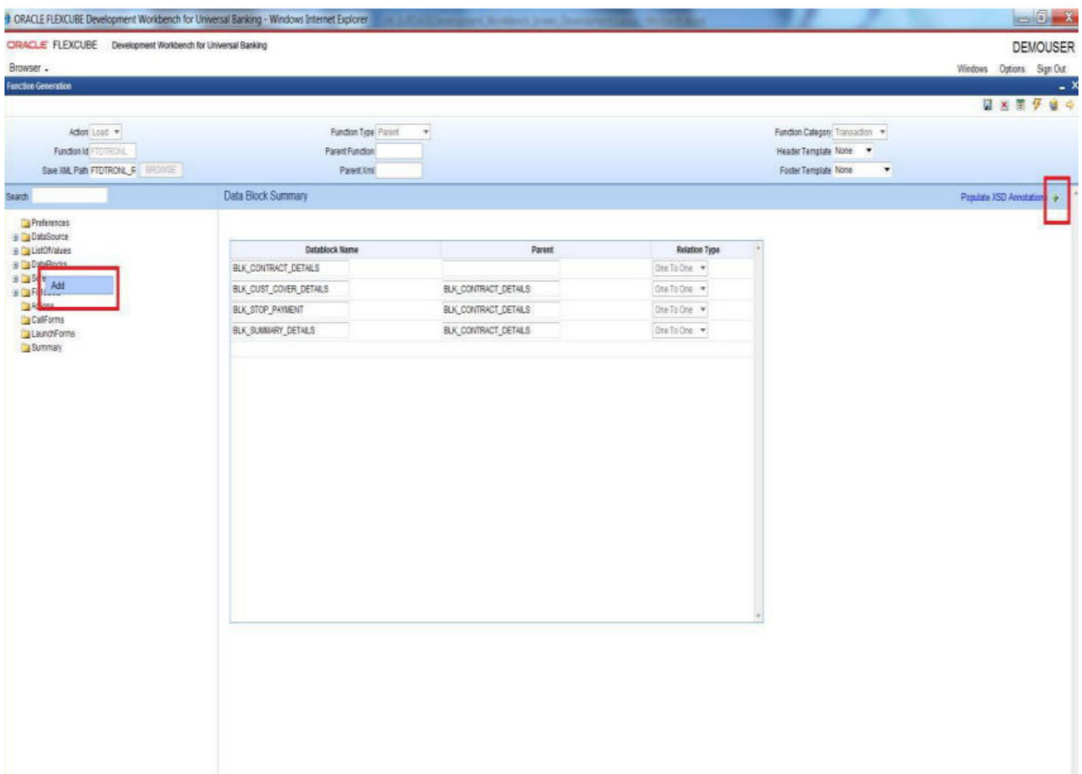
10 Create Data block

This topic provides systematic instructions to create Data Block.

The Data block is a logical grouping of fields from one or more data sources. In general, one data block will have only one data source; but in cases where the data in tables is logically related multiple data sources can be grouped into a single data block. Once the Data source definition is complete, the Block structure of the function id has to be determined based on the functionality of the screen.

1. Right-click on the **Data Block** and select **Add** option to create the data block.
2. Or click the **Add (+)** icon in the top left of the data block screen to create new data block.

Figure 10-1 Adding a New Data Block



Add Block window displays.

3. Specify the **Block Name** and then click on the **OK** button.

   Block Name has to adhere to the following conventions:

   - It has to start with **BLK_**.
   - **Block Name** should not contain the table names which will be attached to the block. This is of high importance from the security point of view as **Block Name** will be exposed in request and response XML.

For example: if the data source name is **FTTB_CONTRACT_MASTER**, the block name can be **BLK_MASTER**; but not **BLK_FTTB_CONTRACT_MASTER**.

4. Specify the Block Properties as follow:

Table 10-1 Block Properties

Field	Desription
Block Name	This field will have defaulted with the block name which is added. It is a non-editable field.
Block Title	Select the Title of the block from the list of labels. If a new label needs to be created, then create and select from the list. For multi-record blocks, the block title will be displayed in the screen preview superseding the Field Set title. For single record blocks, block titles are not of any significance and hence need not be provided.
Parent	Select the parent block from the select list. All the data blocks will be available in the Select list. All data blocks except the master data block need to have a parent data block.
XSD Node	This field captures the XSD node name. ODT will use this name for the generation of the node in XSD's. This field will be defaulted based on **Block Name** (by removing **BLK_** and replacing _ with **-**). If the developer wants, XSD Node field can be modified..
XSD Annotation	This field captures the annotation for the XSD node specified. Description of Block title label will have defaulted as Annotation which can be modified by the developer.
Multi Record	Using this field developer can decide block type, whether the block is a single entry block or multiple entry blocks. Depending on this value, the field **Data sources available** will be populated. If the value is **Yes** the field **Data source available** will be populated with data sources that have flag Multi record **Yes** and vice versa. If the relation type is provided as one-to-one, multi-record has to be **Yes** and vice versa.
Block Pk fields	Primary key fields of the block have to be provided here. If more than one field forms the primary key, it has to be separated by ~. This data is used for building the change log of a screen.
Relation Type	The relation type with the parent data block has to be mentioned here. It can be One To One or One To Many.
Master Block	One master block is mandatory while designing the screen and using this flag you can define a block as master.

Table 10-1 (Cont.) Block Properties

Field	Desription
Block Type	The type of the block can be selected depending on the requirement, below are the description of each type: • **Control**: If the block is used only for UI purpose, and it is not required for processing in the backend, then the block can identified as Control type. Only control fields can be added to the Control block. • **Normal**: This block will be used for normal processing. Request and response XML will contain this data block information. • **Summary**: Block which is used for creating the summary screen. For normal request XML, this block won't be present in the request or response XML.
Data Sources available	Depending on the **Multi Record** flag, data sources will be populated. Then the required data sources can be attached to the block. Validations for populating the data source: • Multi Record: Yes, All the data sources which have **Multi Record** flag **YES** and **Relation Type One To One**. • Multi Record: No, All the data sources which have **Multi Record No**. While attaching Data Sources to Blocks, multiple data sources should be selected only if such data Sources can be functionally clubbed. For Example: CSTM_PRODUCT and LCTM_PRODUCT_DEFINITION. Clubbing should not be arbitrary. A single entry data source can be mapped to more than one data block while a multi-record data source can be attached to only a single data block.
Data Sources Added	This list shows the data sources which can be attached to the particular data block. The data source shown depends on the value of the multi-record field. From the available data sources, the developer can add data sources he wants to be part of this data block. Those will come under the list **Data source Added**. He can also remove them from the data block. The buttons with arrow symbol are used for this addition and removal. Note that one data source can be attached to more than one data block, but each data source column can be mapped to a single data block field. Data Sources added in one release would not be allowed to be removed in any future releases.

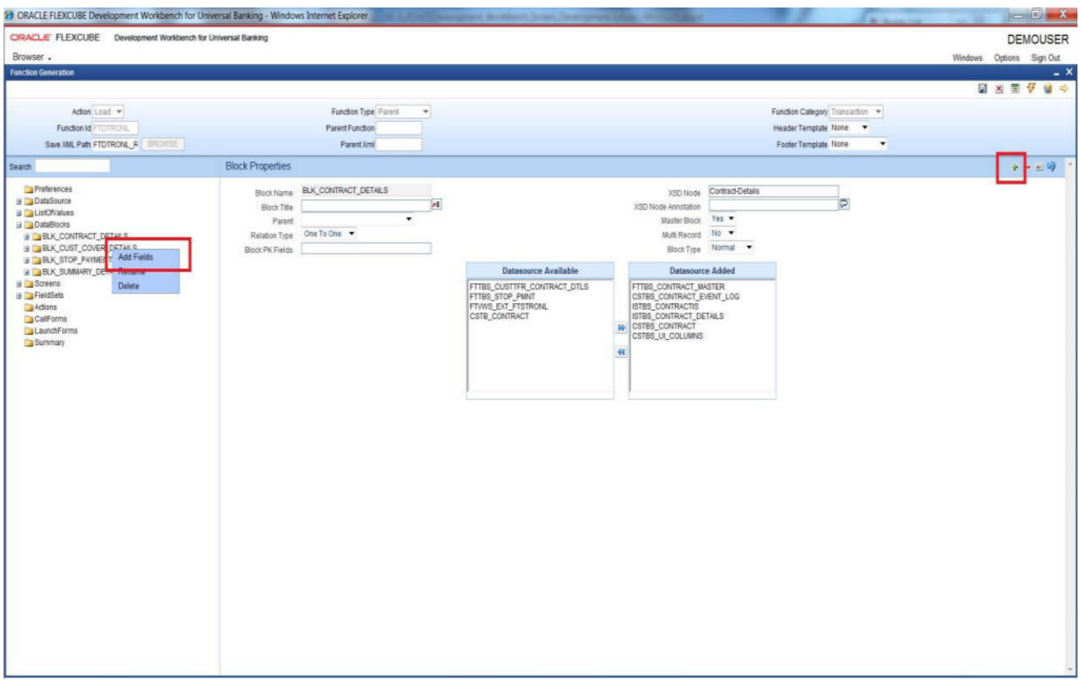
11
Add Data Block Fields to Block

This topic provides systematic instructions to add Data Block Fields to Block.

Data Block Fields are a logical representation of the data source columns in the screen.

1. Right-click on the particular data block and select **Add Fields** option.
2. Or click on **Add Fields (+)** icon on the top left of the data block screen.

Figure 11-1 Adding Block Field to a Data Block



Select Fields & Add UI Fields screen displays.ODT provides an option to add **Data Source Fields** and **UI Fields**.

3. In **Data Source Fields** tab, select all the fields which have to be included in the data block and then click on the **OK** button.
4. In the **UI Fields** tab, specify **Field Name** and click on the **OK** button.

Table 11-1 Data Source & UI Fields

Fields	Description
Data Source Fields	These fields correspond to data source columns created in the radxml. The select list provides the list of all data sources attached to the data block. On selecting a particular data source, all the data source columns are not yet mapped to any block fields. **Field Name** and **Label Code** if maintained in **CSTB_DATA_DICTIONARY** will be defaulted. Otherwise, the developer has to provide the **Field Name** as desired. **Label Code** will be defaulted based on the **Field Name** provided.
UI Fields	**UI Fields** are those which don't have a corresponding data source column. These fields won't be required as part of processing and hence won't be part of request or response XML. Usually, fields like buttons, images, labels, etc are defined as **UI Fields**. A control data block can have only **UI Fields**. Adhere to the following conventions while defining the new block field. • **Field Name** should not be the same as the column name. This is important as the **Field Name** will be visible in request and response XML. • Avoid special characters in the **Field Name** (like the underscore). For example: If the data source column name is CONTRACT_REF_NO, the block field name can be CONTREFNO.

5. Specify the Block Field Properties as follow:

Table 11-2 Block Field Properties

Field	Description
Field Name	The name of the field will have defaulted from the field name provided during the creation of block fields. It is a non-editable field.
Field Label	This has also defaulted during block field creation. This represents the label code for the field. Corresponding label descriptions will be displayed for the particular field. Label code and its description have to be maintained in **CSTB_LABELS**. This can be achieved through ODT or by directly inserting scripts in the business schema. This field is editable. The default label code can be modified, If the developer wishes to change it. LBL code entered manually should be in the below format: **LBL_ xsd tag Name**
XSD Tag	It will have defaulted when a field is added and whenever the label code gets modified. This value is used in the XSD's as **XSD Tag** and the same will be used for web service.
XSD Annotation	Annotation for the XSD tag can be provided here. Description of the label code would have defaulted which can be modified by the developer.

Table 11-2 Block Field Properties

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>Display Type</td><td>Pre-defined values are available in the select box for this field. Value can be selected on requirement. This specifies the HTML display pattern for the field.
Available options are:

Table 11-3 Display Type
<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>Amount</td><td>The display type can be defined as an Amount if it displays an amount. The field will be formatted based on the currency which it picks from the Related Block and Related Field. Related Block and Related Field values have to be mandatorily provided for Amount fields. Related Field should come above the Amount field in the screen and it should hold the currency value for the amount.</td></tr>
<tr><td>Button</td><td>If the Display Type for the field is a Button, the display type has to be selected as a button. Action to be performed on clicking the button should be provided in the Events tab. The button can launch a Sub Screen, Launch form, Call Form or invoke a user-defined JavaScript function.</td></tr>
<tr><td>Checkbox</td><td>The checkbox is used for displaying checkboxes. Attribute Name will be ON and OFF. Attribute values can be provided as per design. In the figure shown below: ON (checked) corresponds to Y in the table and OFF to N. The default value is selected as N.</td></tr>
<tr><td>Date</td><td>If the field is of date type display type can be selected as Date.</td></tr>
<tr><td>Date Time</td><td>If the field displays both date and time, it can be selected as Date Time.</td></tr>
<tr><td>File</td><td>If a Browser option is required for the field, then the field has to be selected as the file type. For Example: If an excel sheet from the local machine needs to be selected, a browser button has to be provided, so that the user can browse the folders and select the file required.</td></tr>
<tr><td>Label</td><td>If the field is required only for displaying an image or Label, it can be selected as Label. Normally UI fields are used for label display purposes.</td></tr>
<tr><td>LOV</td><td>If a field value has to be selected through a LOV, the Display Type has to be selected as LOV. LOV’s</td></tr>
</table>
</td></tr>
</table>

Table 11-2 Block Field Properties

Field	Description
	Table 11-3 Display Type

Field	Description
	have to be defined as explained in the respective section. LOV which has to be linked to the field has to be selected from the LOV Name select list. **Return Fields** and **Bind Variables** have to be mapped as shown in the figure below If the input to the field has to be done only through LOV, **Input By LOV Only** fields can be checked. If validation for the value entered by the user is required in the server, (validating whether the value selected is part of the values fetched by LOV); **LOV validation Required** can be checked. For branch screens, the **Offline LOV Name** can also be attached.
Mask	If the field has to be masked on tab out, then the **Display Type** can be selected as **Mask**. **Mask ID** has to be selected which identifies the pattern of the mask applied.
Password	If the **Display Type** has to be in **Password** form, the display type can be selected as Password.
Radio	If the **Display Type** has to be a **Radio** button, it can be selected as Radio. All the options for the Radio button have to be provided in the **Custom Attributes**. Label which should be displayed in the screen should be provided in attribute name and its corresponding storage value to attribute value. The **Default value** can also be provided.
Read Only Select	This is similar to Select Field, but the field will be non-editable to the user. Select options will be defaulted based on the developer's code.
Restricted Text	This is similar to normal Text field but special characters (non alpha numeric) won't be allowed to be input in a restricted Text field.
Text	For a normal text field, the **Display Type** can be selected as **Text**. This is the default display type.
Text Area	If data to be entered for the field is large (>100 characters); it can be defined as Text Area type. The number of rows and columns required for the text area can be specified in Text Area Rows and Text Area Cols.

Table 11-2 Block Field Properties

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td></td><td>

Table 11-3 Display Type

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>Select</td><td>If a select list is required for a field, the Display Type can be selected as Select. All the options for the Select List have to be provided in the Custom Attributes. Label which should be displayed in the screen should be provided in attribute name and its corresponding storage value to attribute value.</td></tr>
</table>

</td></tr>
<tr><td>Item type</td><td>The value of this field will be defaulted to the DATABASE item if the field has been taken from the data source. If the field is a UI field, the item type can be either CONTROL or DESC. Item type can be provided as CONTROL for non-data fields like buttons, labels, etc. Item Type can be provided as DESC for control fields used for the display of data. For Example: Description of another data source column field, If the type is DESC parent field should be attached and the selected parent field should have a LOV attached to it. And the current field should be one of the return fields of the attached LOV. The field will be generated in the package and front end (SYS.js) only if the Item type is a Database item. CONTROL and DESC types are UI fields; hence won't be present in the generated packages.</td></tr>
<tr><td>Text Area Rows</td><td>This field is applicable only if the Display Type is Text Area. The number of rows required in the Text area can be provided here. This field allows the developer to increase the height of the Text Area on the screen.</td></tr>
<tr><td>Text Area Columns</td><td>This field is also applicable only for Text Area fields. The width of the text area can be provided. This field allows the developer to increase the width of the Text Area on the screen.</td></tr>
<tr><td>Related Block</td><td>This field is applicable if the Display Type is Amount. An amount field can be related to a particular currency field, and this field captures the block name where the currency field is present.</td></tr>
<tr><td>Related Field</td><td>This field is also applicable only if the Display Type is Amount. Subsequent to the above field (Related Block), a field can be selected as a related field for an amount field. Selected Block's (Related Block) fields will be available in the select box.</td></tr>
<tr><td>Parent Field</td><td>If the Item type of the selected field is DESC, a parent field should be attached and the selected parent field should have a LOV attached to it. And the current field should be one of the return fields of the attached LOV.

Note:

This feature is provided to avoid Query data sources and should be used for all description Fields. If a particular field is selected as Description Field in ODT, the LOV for the Parent field should not have any Control Item as a bind variable.</td></tr>
</table>

Table 11-2 Block Field Properties

Field	Description
Minimum Value	The minimum value for a number field can be mentioned. ODT will generate validation for the minimum value of the field in the generated package as well as the front end units.
Maximum Value	The maximum value for a number field can be mentioned .ODT will generate validation for the maximum value of the field in the generated package as well as the front end units.
Maximum Decimals	The maximum Decimals for a number or amount field can be provided here.
LOV Name	This is applicable only if the **Display Type** is **LOV**. LOV to be attached has to be selected from the select list. Select List will contain all the LOVs defined by the developer for the particular screen plus **GLOBAL LOVS**. **Global LOV** can be maintained across function id and the same can be attached to a field. This will reduce development time. LOV can be selected from a list of values attached to the field. This LOV are handled by the FLEXCUBE Infra. **Global LOVs** are stored in **CSTB_LOV_INFO** with function id as **COMMON**.
Offline LOV Name	These LOV will be used by the Branch function id if the branch is in offline mode, offline LOV will be used. All the LOV will be populated which are defined in the LOV definition screen. Both offline and online LOV can be maintained for the same field. **Offline Global LOV**- Defining these LOV remains the same as **Global LOV**, and functionality is the same as offline LOV.
Field set Name	This is a non-editable field and is defaulted based on the field set to which the particular field is mapped.
Data Type	This is a non-editable field. This will have defaulted from the data type of the column to which the field is mapped. This has to be mandatorily provided.
Data Source	This is a non-editable field. This will default the table name of whose column is mapped to the particular field.
Column Name	This is a non-editable field. This will default the column name to which the particular field is mapped.
Maximum Length	The maximum length of a column will have defaulted while adding a field and it can't be overwritten. This value will be considered for the field maximum length of the designed screen. It means, at the run time system will not allow the user to enter the text more than this length. The value of the field can be changed from data source column properties which will reflect in block field level as well.
Field Size	The value will have defaulted while adding the field and the same can be modified. This is the size that will reflect in the screen for the field.
Default Value	A default value can be given to the selected field, the system will use the default value at the run time and the default value will be stored in the generated front end unit, which will not available in the generated packages.
Preview Value	In the Screen Preview, if the developer wishes to see the preview with some data; a preview value can be provided.
Mask Id	This is used for enforcing some restrictions on the values that can be entered in some fields. The ids are present in **sttb_field_mask**. This is applicable only if the **Display Type** is **Mask**.
Popup Edit Required	This can be checked for long text entries. A pop-up edit screen would come along with the field. Pop-up edit will be provided for all fields whose field size is larger than 20 by FLEXCUBE Infra.

Table 11-2 Block Field Properties

Field	Description
Required	If the field is mandatory, this checkbox can be checked. This will force the user to enter a value in this field. **Asterisk (*)** character would come along with the field on the screen.
Visible	If the field is required for processing; but not to be shown on the screen then it can be made invisible.
Input by LOV Only	This is applicable only for **LOV** fields. This will force the user to use the **LOV** button to input the value instead of a key in the value.
Calendar Text	This is applicable if the fields have to be placed close to each other as in a calendar.
Upper Case Only	If the field value has to be in Uppercase only, this option can be checked. This will convert the field value to Uppercase on tab out of that field in FLEXCUBE.
LOV Validation Required	This is applicable only for **LOV** fields. If validation for the value input in the **LOV** field is required, this option can be checked.
Not Required in XSD	If the field is not to be included in XSD, then this option can be checked.
Report Parameter	This is applicable only for Report Screens. If the field is a report Parameter, this option needs to be checked.
Read Only	This will restrict the user from entering a value into the field. Filed can be made non-editable by checking this option.
Custom Attributes	Attribute Name and values for **SELECT**, **RADIO**, **CHECKBOX**, and **READ ONLY SELECT** fields have to be provided here. **Attribute Name** corresponds to the **Label Code** whose description will appear on the screen. **Attribute Value** is the corresponding value that will be stored in the back end for the corresponding attribute. • **Active**: ODT does not allow the deletion of attribute elements if created in an earlier release. Hence if an attribute is no longer required, the attribute can be made inactive. • **Position**: Ordering of the attributes in the list can be manipulated using the **Position** field. Position numbers can be modified. After changing the position values, the position button has to be clicked; which will arrange all attributes in the new order.
Exact fetch	If a field is the **LOV** field, we can check the option of exact fetch required as shown below. If a particular field is attached with an enabled exact fetch value, while searching using this field, the user has to provide the exact value of that field as it is present in the database otherwise the system will show a message of invalid value.
Hot Key Required	If the user wants to provide hotkey functionality to a field then **Hot Key Required** checkbox need to be checked corresponding to that field.
Focus Required	If a field is a read-only field then the focus doesn't come on this field once it is launched in FLEXCUBE. The focus required will bring the focus to the corresponding read-only fields. If the User wants to provide focus required functionality to a field then the **Focus Required** checkbox need to be checked corresponding to that field.

Table 11-2 Block Field Properties

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>Events</td><td>Events tab needs to be input if the field type is BUTTON.

Table 11-4 Events
<table>
<tr><th>Fields</th><th>Description</th></tr>
<tr><td>Event Name</td><td>A pre-defined javascript event can be attached to the field and the same can be selected. Usually, an onclick event is selected.</td></tr>
<tr><td>Function Name</td><td>A function name should be mentioned for the event, the same function should be maintained in the function id javascript or the infra javascript files. This function will be invoked on the event mentioned earlier on the field.

Note:
Function Name needs to be mentioned only if Event Type is either NORMAL or SUBFUNCTION.</td></tr>
<tr><td>Event Type</td><td>Event Type can be:
a. Call Form: If on click of the button a call form has to be launched.
b. Launch Form: If on click of the button a launch form has to be launched.
c. Sub Screen: If on click of the button a Sub Screen has to be launched.
d. OBIEE: If on click of the button the OBIEE screen has to be launched.
e. Normal: If on click of the button, a JavaScript function has to be invoked which is mentioned in the Function Name field. Normal buttons have to be placed in a fieldset for them to appear on the screen.
f. Sub Function: This is used in Process Flow screens to invoke sub-functions.</td></tr>
<tr><td>Button Screen</td><td>It specifies in which screen button is to be placed. This is not required for Normal buttons.</td></tr>
</table>
</td></tr>
</table>

Table 11-2 Block Field Properties

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td></td><td>
Table 11-4 Events
<table>
<tr><th>Fields</th><th>Description</th></tr>
<tr><td>Call Form Name</td><td>If the button is of Call form/Launch form type, the name of the call form/launch form which has to be launched should be mentioned here. All the attached call forms/ Launch forms that are active will appear in the Select List.</td></tr>
<tr><td>Screen Name</td><td>For Sub Screen, Call Form, Launch Form type buttons; this field specified the name of the screen which will be launched.</td></tr>
</table>
</td></tr>
<tr><td>Bind Variables</td><td>Bind Variables field is required for LOV fields. Refer to the section on LOVS for a detailed explanation.</td></tr>
<tr><td>Return Fields</td><td>Return Fields is required for LOV fields. Refer to the section on LOVS for a detailed explanation.</td></tr>
</table>

12

Create New Field Set

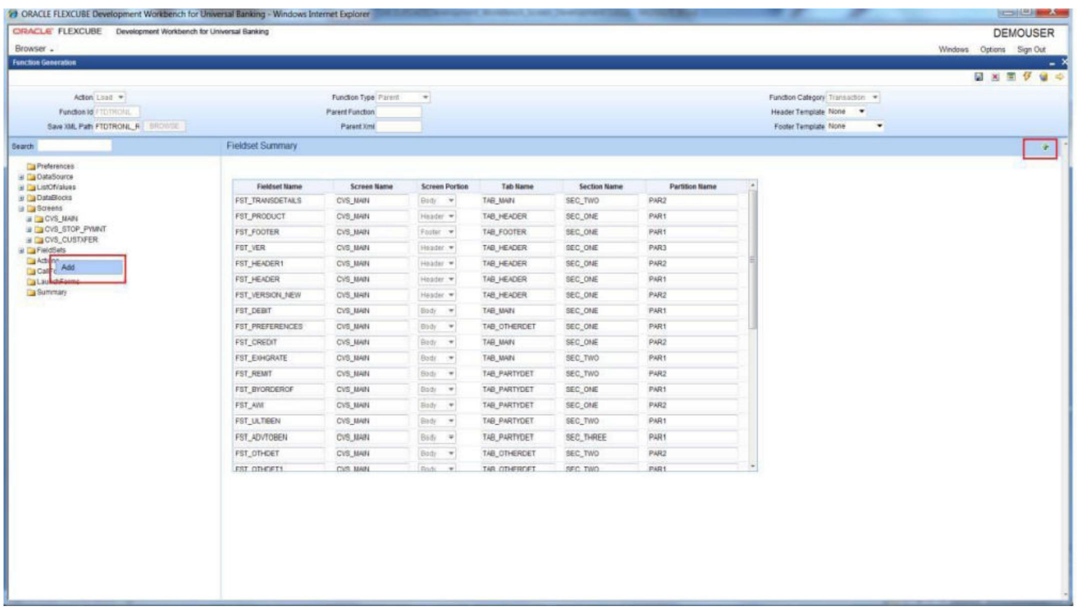
This topic provides systematic instructions to create a new Field Set.

A fieldset is a logical set of fields that would appear together in one partition of the screen layout. The blocks in a Field should be grouped as Fieldsets to make them appear in specific locations on the screen.

1. Right-click on the **Fieldset** Node and select **Add Field Set** option from the right-click menu.
2. Or click on the **Add Fieldset (+)** icon in the top right corner of the Field Set Screen.

   Add FieldSet window displays.

   Figure 12-1 Add Field Set

3. Specify the **FieldSet Name** field and click on the **OK** button.

   The standard naming convention for fieldsets is **FST_< Descriptive Code of the Field Set>**.

   For Example: **FST_VERSION**
4. Specify the details in the **FieldSet Properties** screen as follow:

   Table 12-1 FieldSet Properties

Field	Description
FieldSet Name	This is a non-editable field. Value has defaulted from the fieldset name provided during creation.

Table 12-1 (Cont.) FieldSet Properties

Field	Description
FieldSet Label	All fieldsets can have a title to them, and the same can be mentioned in this field as label code. Label code can be selected using the list of values attached to the field. On the Field Set screen, the Title will appear as the title for a field set.
Data Block	Data block has to be selected from the select list. The select list provides the list of all data blocks in the radxml. The developer can attach fields of the data block selected to the fieldset. **Note:** The data block Name cannot be modified in a future release.
Multi Record	This is a non-editable field. This will be defaulted based on the type of data block selected. If the data block is of type multi-record, the field will be set to **Yes**; otherwise to **No**.
View Type	**View Type** option will be enabled only if the **Multi Record** is **Yes**. If the **Multi Record** is **No**, **View Type** has defaulted to **Single**. If **Multi Record** is **Yes**, the developer can choose the view type as either **Single** or **Multiple** View. In the **Multiple** View type, multi records are displayed in a table grid format. In the **Single** View type, only one record will be displayed at the time. Navigation buttons will be provided for viewing the next record.
FieldSet Height	The developer can provide the height of the fieldset. This is an optional field. FLEXCUBE will calculate the standard height of the fieldset depending on the number of fields and size of the screen etc. Here ODT provides the developer a provision to override the standard settings.
Number Of Rows	If the **View Type** is **Multiple**, then the number of rows in the multiple tables can be decided using the **No of rows** field. This is an optional field, Tool will default to 15 rows if the field value is null.
FieldSet Type	Three options are there: a. **Normal**: This option will be for the normal fieldsets. b. **Image Set**: If a field set is an Image field set then this option needs to be chosen. c. **Version**: If a fieldset represents the version control fieldset then the fieldset type should be version. If Fieldset Type is Version then the Selected Block must have below mentioned fields: i. **VERNO** ii. **LASTVERNO** **Note:** Option for Manual selection of fields will not be provided.

Table 12-1 (Cont.) FieldSet Properties

Field	Description
Screen Name	This field captures the name of the screen where the particular fieldset is to be added. Select List provides the list of all Screens created in the Radxml.
Screen Portion	Select the Screen Portion where the fieldset has to be placed.
Tab Name	Select the tab name where the fieldset has to be placed. Select List will contain all the tabs defined on the screen mentioned above.
Section Name	Select the section in the tab provided earlier to which the fieldset has to be placed. Select List provides all the sections defined in the Tab mentioned above.
Partition Name	Select the partition in the section mentioned above where the fieldset will be placed.
Horizontal FieldSet	A fieldset can be horizontal if this flag is selected. A horizontal fieldset will have fields horizontally arranged, whereas in normal fieldsets all the fields will be arranged vertically by default.
Read Only	This is applicable only fieldset is attached to the multi-record data block. The entire block can be made read-only by selecting this checkbox.
Navigation Button	This is applicable only if **Multi record** is **Yes** and the selected **View Type** is **Single**. If this check box is selected, the tool will add **Previous** and **Next** buttons to the screen. This can be used for navigating multiple records in a single view type.
Visible	If a fieldset is no longer required, it can be made invisible. **Note:** ODT does not allow the developer to delete a fieldset if it is created in a previous release. Instead, the same functionality can be achieved by making it invisible.

13

Rename Screens

This topic provides systematic instructions to rename the screens.

1. Right-click on the **Screen** to be renamed and select the **Rename** option from the right-click menu.
2. Or click on the **Rename** icon in the top right of the particular Screen to rename the screen.

   Renaming of the screen will also be reflected in any reference to the particular screen in the Radxml.

   > **Note:**
   >
   > Renaming of screens would be allowed only if they were created in the same release.

3. Specify the new screen name and click on **Save** icon.

   Screen Name get renamed. Tabs and sections can be renamed similarly.

14 Visible Flag

This topic provides an overview of Visible flag.

If in a future release, if the screen/tab/section needs to be removed, it can be made invisible by unchecking the **Visible** flag. The **Visible** flag is available at Screen, Tab, and Section levels.

Note the following when making a screen invisible:

- If a screen is made invisible, all its tabs and sections will also be made invisible. Any fieldset which is placed on the particular screen will also be made invisible.
- If the same screen is again made visible, tabs and sections still remain invisible. The developer has to manually change the tabs and sections to be visible. Fieldsets also have to be made visible manually. This is done so that if the fields attached to invisible fieldsets were attached to some other fieldset, then the particular fieldset should not be made visible as it will result in more than one fieldset containing the same field.

Note the following when making the tab invisible:

- If a tab is made invisible, all the sections under that tab will also be made invisible. Also, any field set placed in the particular tab will also be made invisible.
- If the tab is again made visible, all the sections under that tab still remain invisible. Fieldsets also remain invisible. The developer has to manually make the section and fieldsets visible as per requirement.
- Tab can be made visible only if its screen is visible. Otherwise, it throws an error.

Similarly, when a section is made invisible, all the fieldsets placed in the section will also become invisible. But when it is again made visible, the fieldsets remain invisible. The section can be made visible only if the corresponding screen and tab are visible.

15

Delete Screens

This topic provides systematic instructions to delete the screens.

1. Right-click on the **Screen** and select the **Delete** option from the right-click menu.
2. Or click on the **Delete** icon in the top right of the particular Screen.

   Any reference to the particular screen in any field set will also be reset to null. Tabs and Sections can also be deleted similarly.

   The selected screen gets deleted.

   > **Note:**
   >
   > ODT will allow deletion of elements only if it is created in the same release. Otherwise, the tool will throw an error as shown below:

16

Screens- Guidelines and Best Practices

This topic provides a list of guidelines and best practices for creating Screens and Tabs.

Below are the practices to be followed while designing screens:

- A function Id should have one main screen.
- Screen Name should start with **CVS_**.
- Every screen will have three portions called **Header**, **Body**, and **Footer**. The developer should not delete these portions.

Keypoints for creating Tabs

Below are the key points to be noted while creating tabs:

- FLEXCUBE architecture does not support multiple tabs in the footer portion of the screen.
- If the screen does not have multiple tabs, then only the **TAB_MAIN** needs to be used. **TAB_HEADER** should not contain any sections in this scenario.
- If the screen is multi-tabbed, **TAB_HEADER** and **TAB_MAIN** should be used while designing. **TAB_HEADER** denotes the header portion of the screen and **TAB_MAIN** should contain the main tab fields.
- **TAB_HEADER**, **TAB_FOOTER**, and **TAB_MAIN** should not be deleted by the developer.
- If any templates are chosen for the footer, **TAB_FOOTER** need not be manually designed.
- Order of Tabs/Sections in the screen can be rearranged by rearranging them in the tree by drag and drop.

17

Multiple Screens in Same Function ID

This topic provides an overview of multiple screens creation in the same function Id.

Multiple Screens can be designed within a single function Id.While launching the function Id from Oracle FLEXCUBE Universal Banking; the main screen will be launched. Buttons have to be placed on the main screen. Button Events to be maintained such that on clicking the button, sub screens will be launched. The button events maintained for the button can be seen in the below figure. The screen name is mentioned as the screen to be launched and the button screen is the screen where the button is placed.

18

Create New Sections and Partitions

This topic provides systematic instructions to create Sections and Partitions for each tab.

Sections have to be created for each tab. The number of sections can vary depending upon the design requirement. All tabs should have at least one section. Partitions should be added to each section. The number of partitions that can be added to a screen depends on the screen size. Partitions get divided into subpartitions.

1. Right-click on the **Tab** and select **Add Section** option from the right-click menu.
2. Or click on the **Add Section** icon in the top right of the Tab screen to create sections for each tab.

   Add Section window displays.
3. Specify the **Section Name** field.

   Section Name should start with **SEC_**.

   For Example: **SEC_ONE**
4. Specify the section details as follow:

Table 18-1 Section Details

Field	Description
Section Name	This field will be defaulted based on the section name provided while creation. It is a non-editable field.
Collapse	If the section has to be made collapsible, this option can be checked.
Visible	If the sections are not required on the screen, they can be made invisible. **Note:** Sections created in previous releases cannot be deleted. Hence it can be made invisible to achieve the same.
Multiple Section	Fieldset attached to this section will allow only multi-entry block fields to be mapped to this fieldset. Mapping for a single entry block will not be allowed if a multiple section option is selected. If multiple section features are to be provided for a section then this option can be selected.

Partitions can be added to the section. The number of partitions allowed on a section depends on screen type and screen portion.

a. For a Large screen, a section in the body or header can have a maximum of 3 partitions.
b. For Large Screens, a section in the footer can have a maximum of 4 partitions.
c. For all medium and small screens, a maximum of 2 partitions are allowed.

The number of partitions mentioned above is including any sub-partitions if defined; that is if the partition is divided into 2 sub-partitions; it will be treated as 2 partitions by the system.

Table 18-2 Partition Details

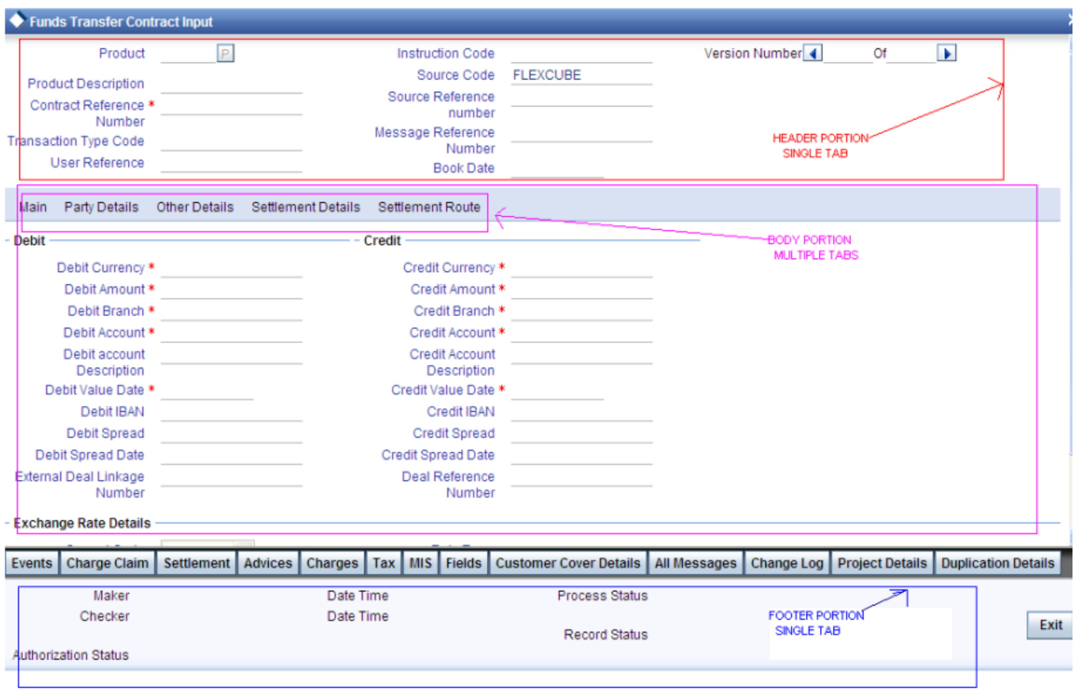
Field	Description
Partition Sl No	This will be used to identify the partition by the system. It will be defaulted by the tool. The numerical value increments by one.
Partition Name	This can be provided by the developer.
Width	The width of the partition will be defaulted by the Tool depending on the number of partitions. For Example: If 2 partitions, the width of each will have defaulted to 50.
Sub Partitions	Each partition can be subdivided into sub-partition provided it does not breach the maximum partitions allowed.

19
Create New Tab

This topic provides systematic instructions to create a new tab.

1. Right click on the screen portion (header/body)/footer) node and select **Add** option from the right-click menu.
2. Or click on **Add** tab icon on the top right of the screen portion screen.

Figure 19-1   Screen Summary



3. Specify the **Tab Name** in the **Add Tab** window.

   Tab Name should start with **TAB_**.

   For Example: **TAB_PREFERENCES**

   Tab Details screen displays.
4. Specify the tab details as follow:

Table 19-1   Tab Details

Field	Description
Screen Name	This is non-editable and will be defaulted from the screen name of which the tab is included.
Tab Name	This is non-editable and it shows the name of the tab provided during the creation of the screen.

Table 19-1 (Cont.) Tab Details

Field	Description
Tab Label	The label Code for the tab can be maintained here. Description of the Label will appear as **Tab Title** on the screen.
Tab Type	**Tab Type** as **Data** or **Service**. Tab Type is Data for Normal case. If any call form has to be embedded in the tab (for example : Branch Screens),this can be selected as Service
Visible	Tabs can be made invisible if it is not intended to be used. **Note:** The tabs created in any previous release cannot be deleted, so the visible flag has to be unchecked to achieve the same.
Dependent Fields	If the **Tab Type** is **Service**, this comes into the picture. This is used to map the fields in the main screen to which this field is dependent.

20

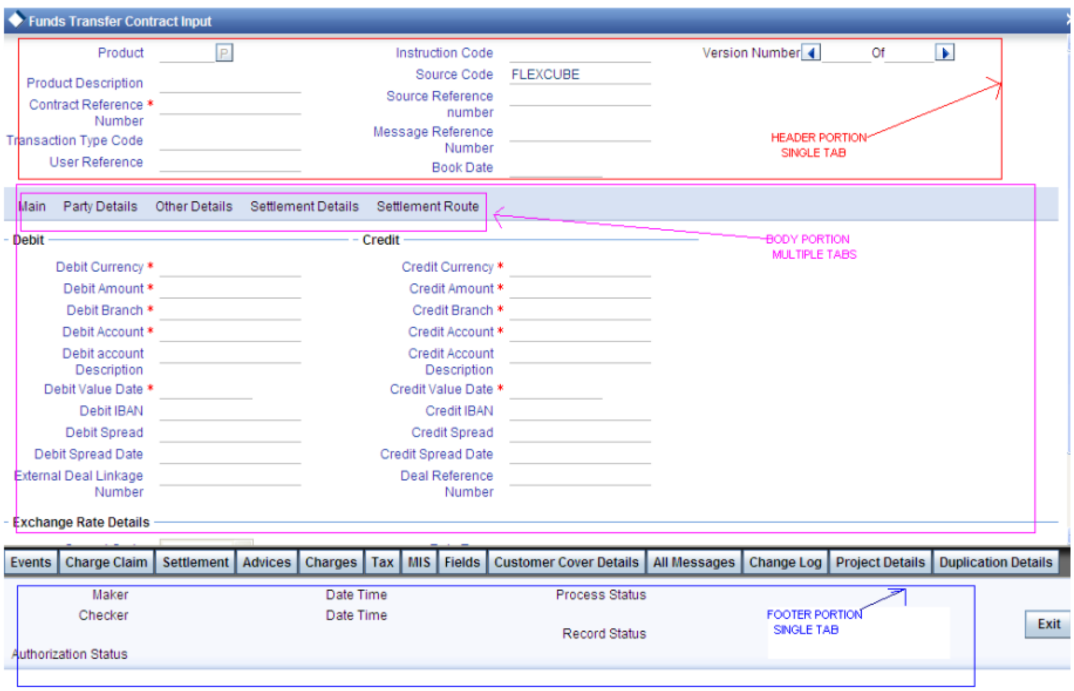
Create New Screen

This topic provides systematic instructions to create a new screen.

After designing Data sources and Blocks, Design the screen layout based on the requirement.

1. Right-click on the **Screen** node and select **Add** option from the right-click menu.
2. Or click on the **Add (+)** icon in the top right of the **Screen** Node screen.

Figure 20-1 Screen Summary



3. Specify the **Screen Name** field.

   Screen Name should start with **CVS_**.

   For Example: **CVS_PREFERENCES**

   Screen Details screen displays.

4. Specify the **Screen Details**.

Table 20-1 Screen Details

Field	Description
Screen Name	This field value will have defaulted when the screen is added and it is a non-editable field.

Table 20-1 (Cont.) Screen Details

Field	Description
Screen Title	This field will have a label code for the screen title to be displayed. Label code can be selected from a list of values button. The label specified will appear in the Title bar of the Screen.
Main Screen	This field identifies whether the screen created is the main screen for the particular function id. FLEXCUBE function id should have one Main Screen. Only the main screen will be launched from the FLEXCUBE when the function id is launched. Other screens have to be launched from the main screen by placing buttons on the main screen.
Visible	Screens can be made invisible if it not intended to be used. The screens created in any previous release cannot be deleted. Hence visible flag has to be unchecked to achieve the same.
OBIEE	If a screen is an **OBIEE** screen then this checkbox needs to be checked. After selecting the screen as OBIEE this screen needs to be attached to some event of a button. It is mandatory to provide the **Screen Title** and the corresponding button label, as both details are required in the UIXML generation.
Screen Size	Depending on the number of fields on the screen, a developer can choose the size of the screen. Options provided are small, medium, and large.
Exit Button Type	This field describes the **EXIT** and **OK** buttons for the screen. These buttons are found in the bottom right corner of the screen. If only the **CANCEL** button is required, select **Default Cancel**. If both **OK** and **CANCEL** buttons are required, select **Default OK Cancel**. If **OK**, **REJECT**, and **CANCEL** buttons are required, select **Default OK REJECT CANCEL**. Example: contract authorization screens

Screen Arguments: Screen Arguments are parameters that can be passed to the screen on launching the screen. If on the launch of the screen, some of the fields need to be populated based on the screen from which it is launched, screen arguments can be used.

Table 20-2 Screen Arguments

Field	Description
Argument Name	This field identifies the name of the argument. This is a mandatory field for a screen argument.

Table 20-2 (Cont.) Screen Arguments

Field	Description
Source Block	This is optional. This is used if the developer wants to pass arguments between two screens in the same function id. The value from one block field (usually present as part of the parent screen) can be passed as a screen argument to another block field (usually present as part of the current screen). Here the parent block field is identified using source block and source field. Source Block identifies the block name in which the source field is present. Select List provides the list of all data blocks added in the function id.
Source Field	Source Field identifies the field name whose value will be passed as the screen argument to the screen. Select List provides a list of all fields in the source block selected.
Argument Value	If the argument value is hardcoded, then this can be specified in this field. If the argument value is specified, then the source block and source field are not required. For example: action codes can be passed as screen arguments which will be hardcoded like EXECUTEQUERY.
Target Block	The target block represents the block containing the field to which the argument value is assigned. This block should be part of the screen. The select list provides the list of all data blocks defined in the function Id.
Target Field	Target Filed identifies the field to which the argument will be passed. Select List provides the list of all fields in the target block.
Active	Screen Arguments defined in an earlier release cannot be deleted. Instead, the developer can make it inactive which serves the same purpose.

21 Rename Data Block

This topic provides systematic instructions to rename the Data Block and Block field.

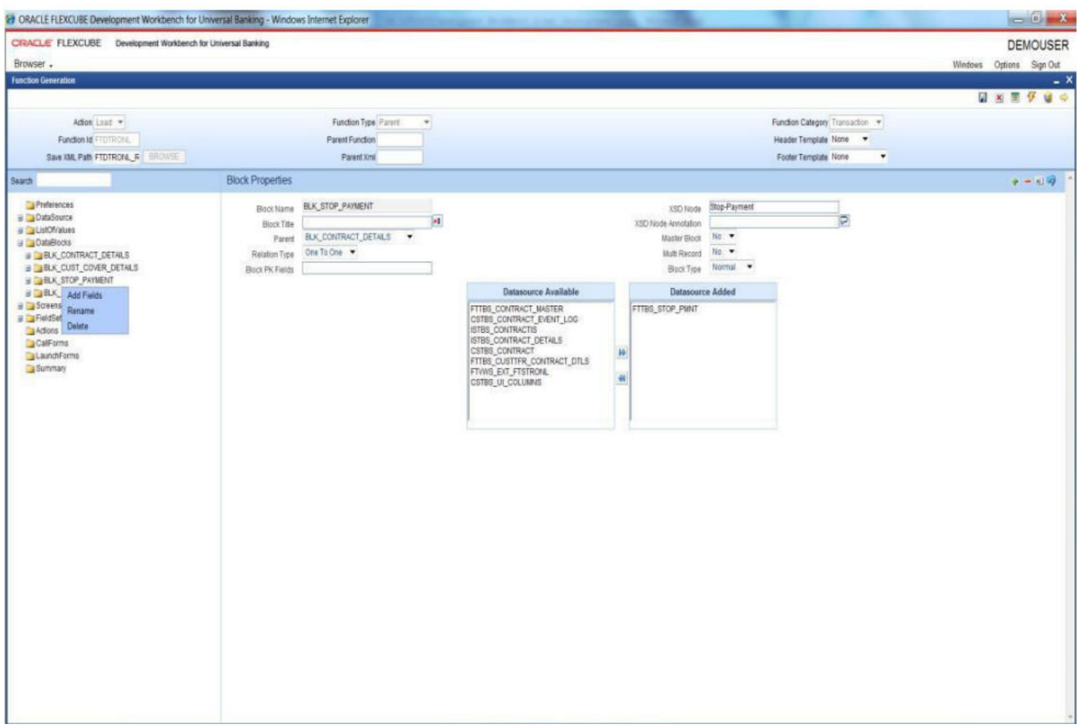
1. Right-click on the data block to be renamed and select the **Rename** option from the right-click menu.
2. Or click on the **Rename** icon on the top right portion of the data block screen to rename the Data Block.

   Renaming of data block will also rename all the references to the data block. For Example: Summary data block if renamed will reflect in the summary node also

   > **Note:**
   >
   > ODT will allow renaming the data block only if it is created in the same release. Data blocks created in any previous release cannot be renamed.

   The **Block Properties** screen displays to enter new clock name.

   Figure 21-1 Block Properties



Renaming Of Block Field: The Block field can also be renamed provided it satisfied the condition above. The process is similar to the renaming of data source columns. All the references to the renamed block field will also be renamed. Since the renaming of data block or block fields is not allowed in later releases, it is advised to take proper care while designing data blocks for the screen.

Note:

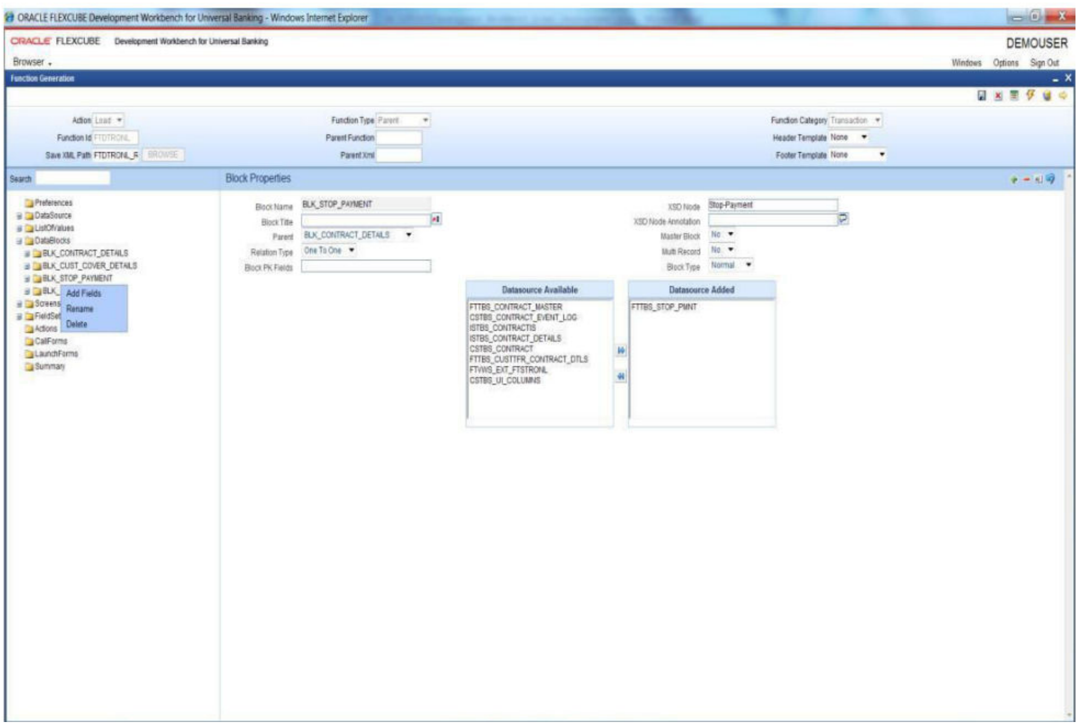
All the units need to be regenerated after renaming a data block or a block field.

22
Delete Data Block

This topic provides systematic instructions to delete the Data Block and Block Field.

1. Right-click on the data block to be deleted and select the **Delete** option from the right-click menu.
2. Or click on the **Delete** icon in the top right of the particular data block screen to delete the data block.

Figure 22-1 Block Properties



Deletion of data block will also delete all the references to any of its block fields. It may remove any of its fields attached to a fieldset, or the block fields used in the summary screen.

> **Note:**
>
> ODT will allow deleting the data block only if it is created in the same release. Data blocks created in any previous release cannot be deleted.

The selected data block gets deleted from the **DataBlocks** node.
Deletion of Block Field: The Block field can also be deleted provided it satisfied the condition above. The process is similar to the deletion of data source columns. All the

references to the deleted block field will also be deleted. Since deletion of data block or block fields is not allowed in later releases, it is advised to take proper care while designing data blocks for the screen.

> **Note:**
>
> All the units need to be regenerated after the deletion of a data block or a block field.

23

Data block- Guidelines and Best Practices

This topic provides a list of guidelines and best practices for creating data blocks.

Below are points to be noted while creating data blocks:

1. All Block Names Should start with **BLK_** and the remaining portion of the block name should be the same as the XSD node of the Block Name in Upper case and _ replaced with **-**. Avoid naming the block name the same as that of the table name.
2. Blocks also would have a hierarchy and has to be chosen correctly.
3. Block Types are similar to data source types with an additional type **Control**.
4. After adding a block, ODT defaults the XSD node. Node name would be Block name without **BLK_** and in Sentence case. Also, _ will be replaced with **-**.
5. Parent block should be chosen from the list based on the hierarchy.
6. Similar to data sources, block also can be either Multi-Record or single record type.
7. Block Title has to be selected from the LOV.
8. Relation type has to be mentioned as **One to One** or **One to Many**.
9. One and only one block should be selected as Master Block.

Keypoints for attaching Data Sources to Data Block

Below are some of the important points to be noted while attaching data sources to a data block:

1. The same data source can be selected in multiple blocks. However, in the case of Multi record data sources, ODT allows the developer to attach the data source to only one block.
2. One Multi Record block can have multiple data sources provided all of these data sources have strict one-to-one relation.
3. In case of multiple data sources in a single data block, it is important to ensure that the data in the attached data sources is functionally related and more importantly, a row in one table would definitely have a row in other attached data sources as well. Otherwise the generated code inserts records in all the tables always and this would result in lot of dummy records in the database.
4. Parent data block has to be above its child data block in the browser tree. Developer has the option to rearrange the order of the blocks in the tree (by drag and drop).

> **Note:**
>
> If block or block fields are rearranged, all units will have to be regenerated.

Keypoints for specifying Field Properties

Below are some of the important points to be noted while specifying field properties:

1. Avoid using UI fields as LOV bind variables.

2. Avoid using separate Query data sources for single description fields. Use Description type UI fields for Description fields.
3. Ensure that Related Block and Field are given for Amount Fields. Ensure that related currency fields are placed above the amount fields in the browser tree.
4. In case the field is not required in XSD, check not Required XSD. There are cases where the field is a primary key field in a child block and does not need to be repeated in Child block. For example, the field PRODUCT_CODE in Product forms is not required in any Block other than once in the master block.

24
Field Sets- Guidelines and Best Practices

This topic provides a list of guidelines and best practices for designing Field Sets.

Below are the guidelines to be followed while designing Field Sets:

1. Ensure that proper Label Code is given a title and make sure that the same is available in **CSTB_LABELS**.
2. Select the Screen, Section, and partition and the Block from which the fieldset is to be created.
3. Select the fields into the fieldset from the left text area (data block fields) to right and give the appropriate Sub partition number wherever applicable.
4. In case the fieldset has to be horizontal, check the **Horizontal Fieldset** check box.
5. For multi-record fieldsets (either single view or multiple entry view), check the **Read-only** check box to avoid **+/-** buttons.

> **Note:**
>
> Checking **Read-only** check box would only stop the user from adding and deleting records. The system would still allow modifying the fields of a row based on whether or not they are read-only.

25
Attaching Field to a Field Set

This topic provides an overview of the process of attaching Field to a Field Set.

Once the data block and the screen details (up to partition) in which fieldset has to be placed are identified, fields can be attached to the fieldset.

Data Block Fields

This List Select box provides all the block fields of the data block mentioned which is not yet attached to any visible fieldset. Any of the fields available in the data block fields can be moved to the fieldset fields by selecting and clicking on the move arrows. Similarly, any attached field can be removed from the fieldset by moving it back to data block fields.

Field Set Fields

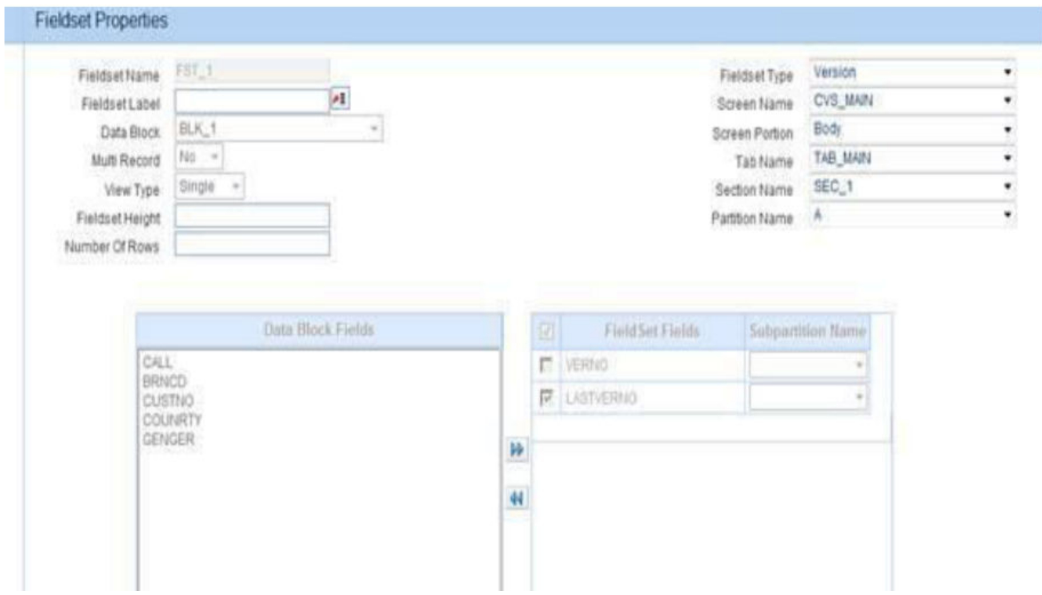
This contains the list of all fields which is attached to the fieldset. The order of the fields in this grid signifies the order in which it will be displayed on the screen. Fields can be re-arranged as per requirement. If the partition mentioned in the fieldset properties is divided into sub-partitions, then the subpartition in which the particular field has to come has to be selected from the subpartition list.

26 Delete Field Set

This topic provides systematic instructions to delete the Field Set.

1. Right-click on the **FieldSet** node and select the **Delete** option from the right-click menu.
2. Or click on the **Delete FieldSet (-)** icon in the top right corner of the fieldset screen.

Figure 26-1 Field Set Properties



> **Note:**
>
> The deletion of the fieldset is allowed only if the fieldset is created in the same release.

The selected FieldSet gets deleted from the **FieldSet** node.

27

Visible Flag- FieldSet

This topic provides an overview of visible flag.

If the fieldset is no longer required in any future release, then the fieldset can be made invisible. This serves the same purpose as deleting the fieldset.

Note the following while making field set as invisible:

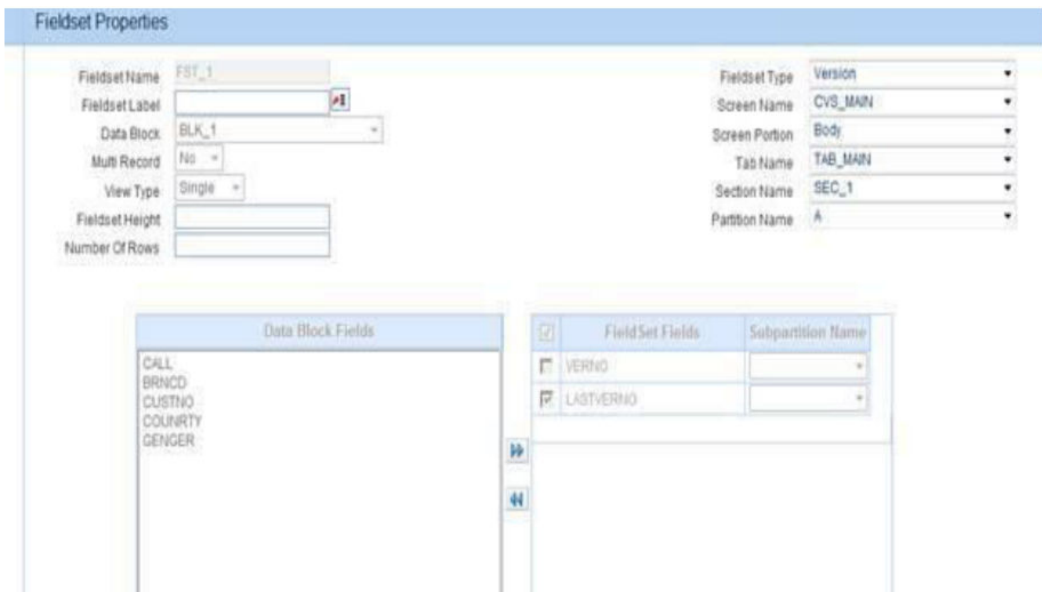
- When a fieldset is made invisible, all the fieldset properties will be disabled. The developer won't be able to add any new field to an invisible field set, but he will be allowed to remove the field from the fieldset fields.
- If he makes the fieldset visible again, the System validates whether the Screen, Tab, and section in which the fieldset is placed are visible. If any of them is not, the system throws an error.
- When the fieldset is made visible, the system checks whether any of the fields attached to this fieldset has been re-used in any other fieldset. If any fields are found to be attached to any other visible fieldset, it won't allow making the fieldset visible.

28
Rename Field Set

This topic provides systematic instructions to rename the Field set.

1. Right-click on the Field Set to be renamed and select the **Rename** option from the right-click menu.
2. Or click on the **Rename Field Set** icon in the top right corner of the Field Set screen.

Figure 28-1 Field Set Properties



> **Note:**
>
> Renaming of the fieldset is allowed only if the fieldset is created in the current release.

3. Specify the new **Fieldset Name** and click on **Save** icon.

   The Field Set get renamed.

29 LOVs

This topic provides an overview of the List of Values (LOVs).

FLEXCUBE supports two types of List of Values (LOVs):

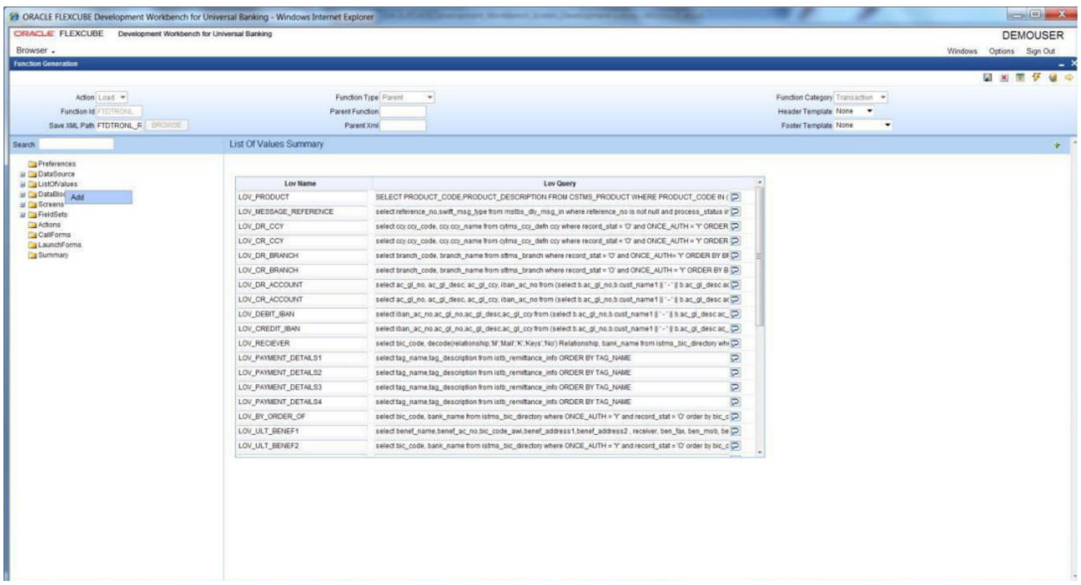
1. **Global LOVs**: These are LOVS defined in the system which can be used across all the functions. Global LOVs are stored in **CSTB_LOV_INFO** with function id as **COMMON**.
2. **Local LOVs**: These are defined in particular to the function Id. They are defined in the function Id and can be attached to any field in the function Id as per requirement.

30 Define LOVs

This topic provides systematic instructions to define LOVs.

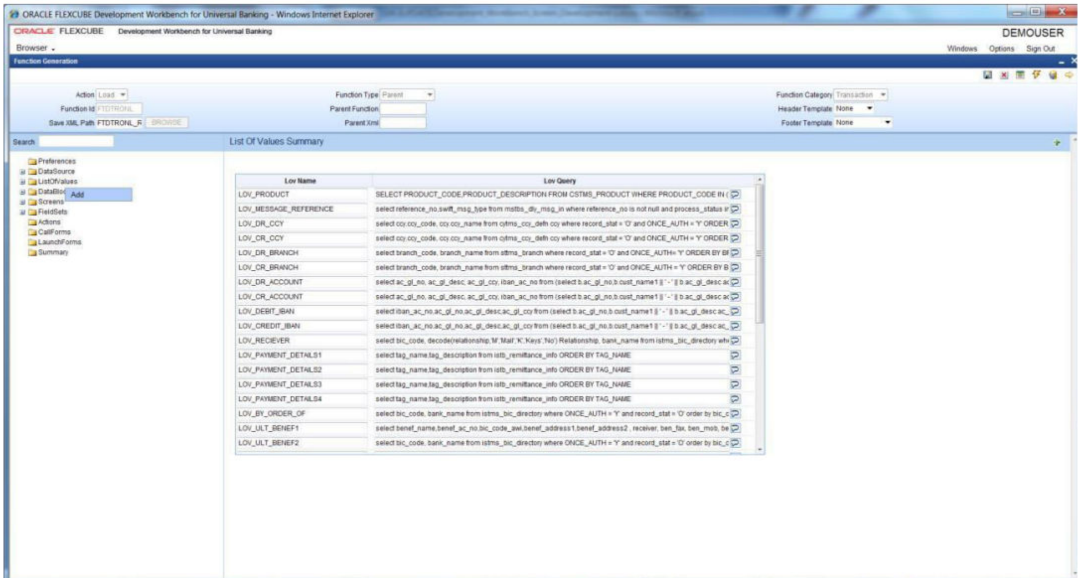
1. Right-click on the **ListOfValues** node and select **Add** option from the right-click menu.
2. Or click on the **Add (+)** icon in the top right portion of the LOV grid screen.

Figure 30-1 List Of Values Summary



LOV window displays.

Figure 30-2 List Of Values Summary



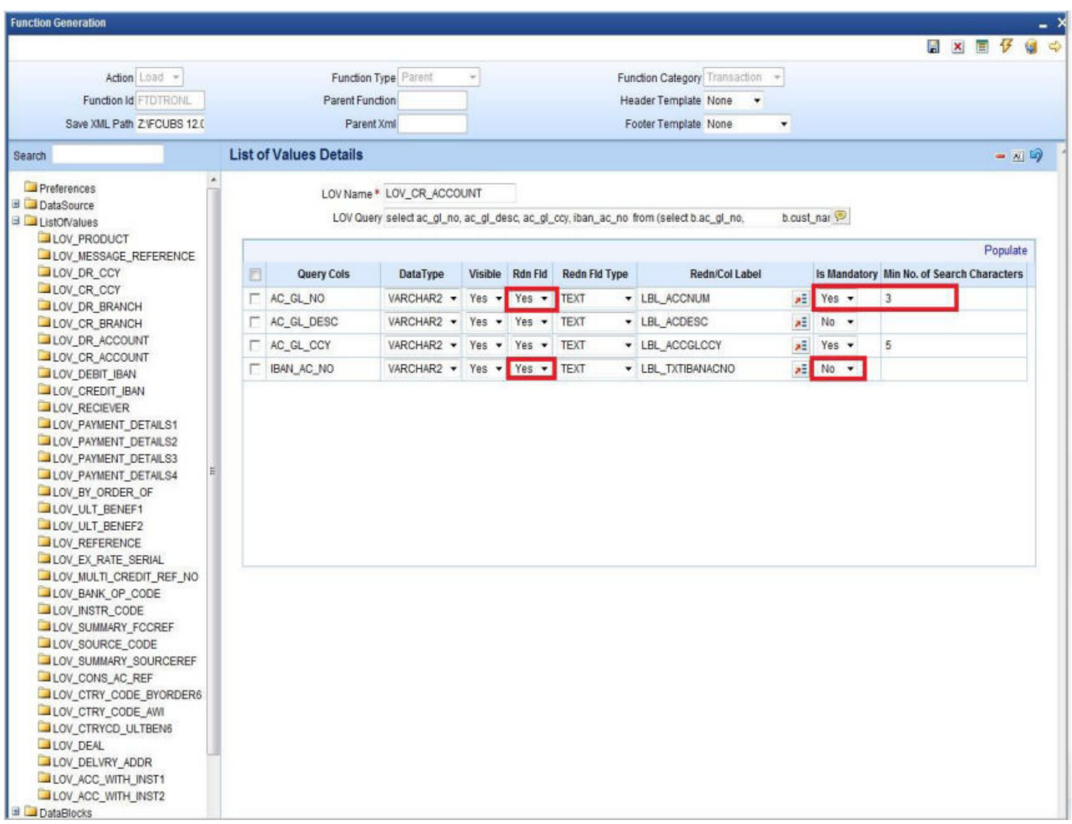
3. Specify the **LOV Name** field and then click on **OK** button.

   LOV name should start with **LOV_**.

   For Example: **LOV_COUNTRY**

   List Of Values Details screen displays.

Figure 30-3 List Of Values Details



4. Specify the fields in the **List Of Values Details** screen.

Table 30-1 List Of Values Details

Field	Description
LOV Name	This is a non-editable field. It will be defaulted based on the LOV name provided while creating LOV.
LOV Query	LOV query has to be provided in this field. **Bind variables** for the LOV have to be specified as **?**. Bind variables are the parameters whose values are required as input for executing the query during run time. For Example: SELECT CUSTOMER_TYPE FROM STTM_CUSTOMER WHERE CUSTOMER_NO=? Here value for Customer No is to be passed as a parameter.

Table 30-1 (Cont.) List Of Values Details

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>LOV Column Details</td><td>Click on Populate button. This will default all the columns which will come in the output of the LOV query. For Example: for the above query, it will populate only one row with Query Cols value as CUSTOMER_TYPE.

Table 30-2 LOV Column Details
<table>
<tr><th>LOV Column Details</th><th>Description</th></tr>
<tr><td>Query Cols</td><td>This is a non-editable field. This will be populated by the system on clicking of Populate button. All the columns from the query result will be populated as query cols.</td></tr>
<tr><td>Data Type</td><td>This row will also be defaulted based on the LOV query. The data type of the query cols has to be provided here.</td></tr>
<tr><td>Visible</td><td>The Result Column of the LOV can be made invisible by specifying in this column.</td></tr>
<tr><td>Reduction Fld</td><td>A field can be made as a reduction of non-reduction by using this flag. If the field is a reduction field, then in the LOV screen in FLEXCUBE, the user will have to filter the list of values based on the reduction fields.</td></tr>
<tr><td>Reduction Fld Type</td><td>The display type of the reduction field can be specified here. Display type of the reduction field can be either of the below-given values:
• TEXT
• CHECK BOX
• RADIO
• SELECT</td></tr>
<tr><td>Redn/Col Label</td><td>The reduction fields should be provided with proper labels so that their description comes on the LOV screen. Label Codes have to be mandatorily maintained for the functioning of the LOV screen.</td></tr>
<tr><td>Is Mandatory</td><td>If the reduction field id is selected as YES, ODT will allow selecting the Is Mandatory option for a</td></tr>
</table></td></tr>
</table>

Table 30-1 List Of Values Details

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td></td><td>

Table 30-2 LOV Column Details

<table>
<tr><th>LOV Column Details</th><th>Description</th></tr>
<tr><td></td><td>particular column. If Is Mandatory is selected as Yes, ODT will ask for minimum search character length. By default, the value is 3. If the field is mandatory and minimum search character length is given as n, then while searching, the user has to enter a minimum of n characters to search a particular value otherwise the system will show an error message.</td></tr>
<tr><td>Minimum Search Character length</td><td>If Is Mandatory is selected as YES, then only ODT will allow entering values for Min search character length. By default, the value is 3. Any value less than 3 will not be accepted. ODT will show an alert message in case of any rule violation. Once the value for the Minimum search character length field is provided, the user has to enter minimum character length to fetch a value corresponding to that return field.</td></tr>
</table>

</td></tr>
</table>

LOVs have to be attached to the block field as per requirement.

5. In the block field, select the **Display Type** as **LOV** to attach LOV to the Block Field.
6. Specify the below given details in the Block Field Properties screen:

Table 30-3 LOV Details

Field	Description
LOV name	Select List will contain both **Local LOVs** and **Global LOVs**. The developer has to select the LOV as required for the field.
Input By LOV Only	This field has to be checked if the field has to be input through LOV only.
LOV Validation Reqd	If validation for the entered value is required against the values fetched from the LOV query, this checkbox can be checked.

Table 30-3 (Cont.) LOV Details

Field	Description
Bind Variables	Bind variables defined in the LOV query have to be mapped to the corresponding data block field. During LOV query execution, the value of these data blocks will be picked up in place of bind variables (?). Click on the button **Default from LOV Definition**. The number of rows corresponding to the number of bind variables in the query will be created. Below details has to be maintained in the bind variable tab: • **Block Name**: The block which contains the bind variable block field has to be selected from the list of all data blocks. • **Bind Variable**: The block field which is the bind variable has to be selected from a list of fields of the block selected. • **Data Type**: The data type of the bind variable has to be mentioned here. This can be **STRING**, **DATE**, **NUMBER**, etc. If more than one bind variable is present in the LOV query, bind variables have to be provided in the same order as it appears in the LOV query.
Return Fields	Information regarding the return fields has to be provided in the **Return Fields** Tab. The developer has to map the block fields to which the selected values from LOV should be assigned. Click on the button **Default from LOV Definition**. All Query Cols will be defaulted based on the LOV definition. • **Query Column**: This will be defaulted by the system based on the LOV definition on clicking **Default** button. • **Block Name**: Provide the Block Name which contains the field to which the particular coulmn value has to assigned. • **Return Fld Name**: Select the data block field to which the result will be assigned. All the query Cols need not have a return field.

In the figure below **AC_GL_NO** is mapped to the CRACC field. Therefore the value of the result column **AC_GL_NO** from the selected record will have defaulted to the CRACC field.

LOV Details will be generated in the script for **CSTB_LOV_INFO** which needs to be compiled in the FLEXCUBE schema for the functioning of LOVs.

31 LOVs- Guidelines and Best Practices

This topic provides a list of guidelines for defining LOVs.

Note the following points while defining LOVs:

- Avoid using in-line views.
- Avoid order by clause, as the same can be chosen by the user at run time.
- Select proper Label Codes and ensure that the same are available in **CSTB_LABELS**.

32

Delete and Rename LOVs

This topic describes the process of deletion and renaming LOVs.

Deletion and Renaming of LOVs can be done provided they are created in the same release. The procedure for deletion/renaming is similar to those of data blocks, Refer below given topics:

- #unique_59
- #unique_60

33
Call Forms

This topic provides an overview of Call Forms.

Call forms are function IDs that do processing which is common across many screens. Call forms cannot be launched independently. They need to be launched from another function Id. The third letter of the function id will be C for a call form. Most of the subsystems of contract screens are designed as call forms.

For Example: Settlement Screen, tax details, etc.

- Call Form- Guidelines and Best Practices
  This topic provides a list of guidelines and best practices for attaching the Call Form to the Screen.

33.1 Call Form- Guidelines and Best Practices

This topic provides a list of guidelines and best practices for attaching the Call Form to the Screen.

Note the following points while attaching the call form to the screen:

- A maintenance Call form should be attached only to a maintenance function. Similarly, only transaction call forms can be attached to a transaction screen.
- Static Scripts for **CSTB_CALL_FORM_NODES** and **SMTB_MENU** generated by the Call Form Radxml has to be compiled in the FLEXCUBE schema before attaching that call form to any screen. Call form details are populated based on the maintenance in these tables. Therefore ensure that data is present in **CSTB_CALL_FOR_NODES** and **SMTB_MENU** for all the attached call forms.

34
Launch Forms

This topic provides an overview of Launch Forms.

Launch Forms are function ids which can be launched from another screen for data view purpose. No processing can be done on Launch Form screen data as done in Call form. Launch Form is like any other normal function id and it can be launched independently.

For Example: Screen for viewing accounting entries for a transaction

- Attach Launch Forms
  This topic provides systematic instructions to attach Launch Forms.
- Launch Forms- Guidelines and Best Practices
  This topic provides a list of guidelines for attaching Launch Forms.
- Summary- Guidelines and Best Practices
  This topic provides a list of guidelines for designing the summary screen.

34.1 Attach Launch Forms

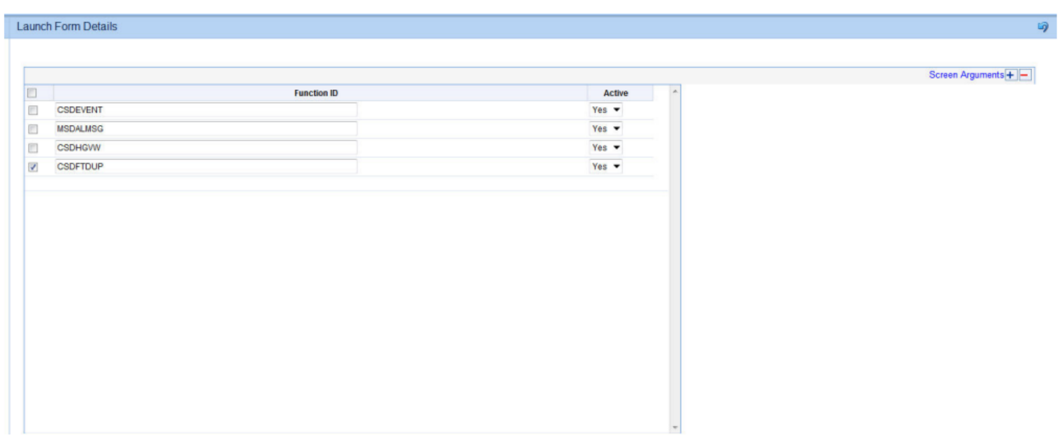
This topic provides systematic instructions to attach Launch Forms.

1. Click on the **Add (+)** icon in the **Launch Form Details** screen to add Launch form.

   Launch Forms to be attached to a function Id has to be maintained in the Launch Form node.

   Launch Form Details screen displays.

   Figure 34-1 Attaching Launch forms to a Function Id

2. Specify the details in the **Launch Form Details** screen.

Table 34-1 Launch Form Details

Field	Description
Function ID	The name of the Launch Form function id to be attached has to be specified here.
Active	Launch Forms attached in one release cannot be removed in a future release. Instead, the active flag can be set as **NO** to achieve the same.
Screen Arguments	Launch Form is used for query purposes only. Thus primary key and the action code have to be passed on to the Launch form screen on launching Launch Form. Select the **Launch Form** and click on the **Screen Arguments** button. The following screen will be launched. Click **Populate** button to default the screen arguments. **Note:** The Screen arguments for the Launch Form have to be maintained in the **CSTB_CALL_FORM_NODES** table. • **Argument Name**: This will be defaulted based on the screen arguments specified for Launch Form main screen maintained in **CSTB_CALL_FORM_NODES**. • **Source Block & Source Field**: Provide the block Name and source field of the function id whose value will be passed to the screen argument mentioned. • **Argument Value**: If the screen argument value is to be hardcoded, then the value can be mentioned in this field. In this scenario, source block and source field need not be mentioned.
Attaching Launch form to Main Screen Using Button	Launch forms can be launched by clicking on the button placed on the main screen. Button events have to be maintained such that Launch Form will be launched on clicking it.

34.2 Launch Forms- Guidelines and Best Practices

This topic provides a list of guidelines for attaching Launch Forms.

Note the following points while attaching Launch Forms:

- Entry has to be made manually in **CSTB_CALL_FORM_NODES** for the launch form. The script won't be generated by the Tool while designing the Launch Form. Hence it has to be inserted manually providing the screen arguments as maintained for Launch Form main screen.
- Launch Forms are used only for querying data. Hence **ACTION_CODE** has to be passed as a screen argument with argument value as **EXECUTEQUERY** and the Primary Key values for querying in the launch Form screen should be passed as the other parameters.

34.3 Summary- Guidelines and Best Practices

This topic provides a list of guidelines for designing the summary screen.

Note the following points while designing the summary screen:

- Maximum number of Query Fields allowed for a summary screen is 12.
- For the Maintenance screen, if the summary data block and master data block are not the same, care should be taken so that the names of the primary key fields of both the blocks should be the same.

35 Create Action Screen

This topic provides systematic instructions to create the Action screen.

1. Under the **Function Generation** screen, click on the **Action** node.

   Web Service related information and FLEXCUBE amendable fields details are captured in the actions screen.

2. Specify the details of the **Form Actions** screen as follow:

Table 35-1 Form Actions Details

Field	Description
XSD Type Identifier	A unique descriptive code should be given in this field, which would describe the nature of the Function. For example **LCProd**, For LC Product Definition Form Screen Layout Design
Service Name	An appropriate Web service name should be selected here. LOV will fetch the service names maintained in **GWTM_OPERATIONS_MASTER**.
Operation Id	This is the key using which the Web service operation code would be derived. For example of Operation Id is **Product** Operation Code for **QUERY** would be **QueryProduct**. Note that this should be given in the Sentence case and should be unique within the service. Ensure that the correct operation id is given that is in the case of LC Product Operation id can be **Product**. It need not be **LCProduct**. In case of multiple products/contracts there under the same service, for example in Exchange-traded derivates module, Operation Id for Deal product can **DealProd** and for Margin product **MarginProd** etc.
Action Code	Pre-defined actions will be available for the action code. These are the operations that are available in the FLEXCUBE. For maintenance, the type of function system enables the below action codes only: • **QUERY** • **NEW** • **MODIFY** • **AUTHORIZE** • **DELETE** • **CLOSE** • **REOPEN** • **SUMMARYQUERY** For transaction Screens, all the actions would be enabled.

Table 35-1 (Cont.) Form Actions Details

Field	Description
Operation Code	If the web service is selected, the operation code will have defaulted and the operation code will be the combination of action code and Operation Id and the same operation code will be used to perform operations for web service. If the developer wants custom-specific Operation Code names, they can be provided so. For this double click on a particular action code, the **Operation Code** field for that action code will be enabled. Developer can modify the operation code now.
Action Stage Type	For multistage actions, it should be checked. If any field has to be made amendable during the particular operation, the action stage type has to be checked for the particular action.

3. Click on the **Amendable** button against each action to capture the amendable information for each action.

   Amendable button against each action would capture the amendable information for each action. It is mandatory to give amendable information for applicable actions for web services to function normally.

   Amendable Details screen displays.

4. Specify the details in the **Amendable Details** screen for each action.

   Amendable information is not just for **Modification** and for all the applicable actions. For example in a transaction screen, for QUERY action amendable information could be **Oracle FLEXCUBE Reference number**, **User Reference Number**, and **External Reference Number**. It is not necessary to just have the Primary key in the XSD as external systems might query contracts based on any of these reference numbers. At a block-level user has to specify **All Records**, **Delete Allowed**, **New Allowed**, and **Mandatory**. This information is used by the Tool to generate scripts and code accordingly.

   Table 35-2 Amendable Details

Field	Description
All Records	If All records are checked, System expects the full data every time; that is in case of a multi-record block if there are 10 records and 1 record has to be modified, external systems still should send the remaining 9 records with the old data.
New Allowed	This indicates whether a new record can be added or not, in this block as part of this action.
Delete Allowed	This indicates whether a record can be deleted or not, in this block as part of this action.
Mandatory	This flag indicates whether the node is mandatory to be sent from external systems. This flag would be used to set the Min Occurs Flag in XSDs for that operation.

   Generated code also would perform validations based on the above preferences.

5. Click on the **OK** button.

36
Design Summary

This topic provides systematic instructions to design the Summary screen.

1. Under the **Function Generation** screen, click on the **Summary** node.

   The summary screen can be designed using ODT if applicable.

   Summary Details screen displays.

   Figure 36-1 Summary Details

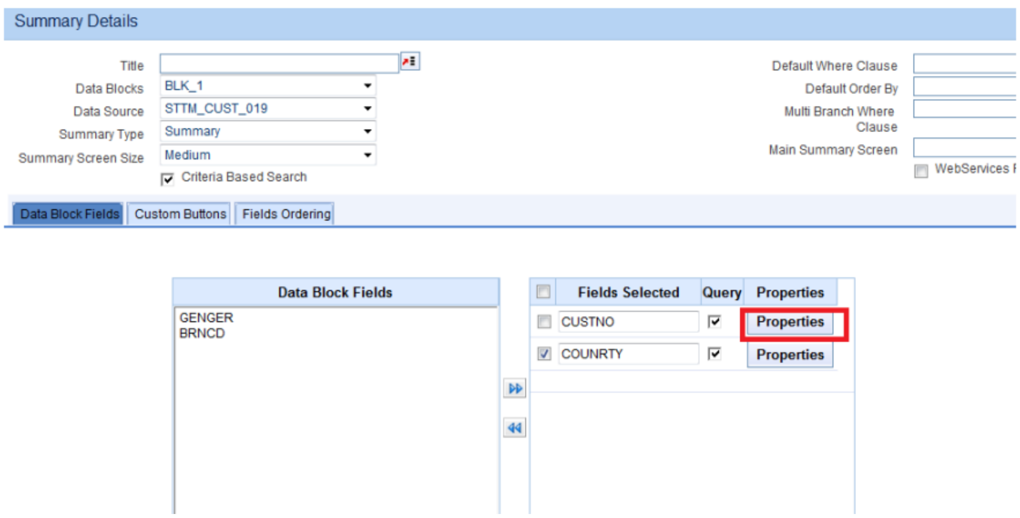


   Figure 36-2 Summary Details

2. Specify the details in the **Summary Details** screen.

   Table 36-1 Summary Details

Field	Description
Title	Summary screen title can be maintained here using a label code, and label code can be selected from the list of values attached to the field.
Data Blocks	A block has to be selected as **Summary Block**. This block can be either one of the Single record blocks of the function or a block of type Summary. In most cases Master Block itself can be used for Summary also and if the requirement is to show some other information a new summary type block can be created and can be used.

Table 36-1 (Cont.) Summary Details

Field	Description
Data Source	Captures the data source name of the summary, on which query should happen for summary. This will also be the data source attached to the summary data block.
Summary Type	Summary Type can be of: • **Summary**: This is the default option. For a normal summary screen to a function id, this option can be used. • **Query**: This is defunct. • **Bulk Authorization**: Checkbox option will be provided in the Summary screen. The user will have the option to select multiple records in the summary screen at the same time and process them. For Example: Bulk authorization screen • **Upload**
Summary Screen Size	This field captures the size of the summary screen.
Default Where Clause	This field captures the default where clause for summary query.
Default Order By	This field captures the default order by clause for the summary query.
Multi Branch Where Clause	This screen is applicable only if multi-branch access is checked for the screen in the **Preferences** node. This field captures the summary where clause for multi-branch screens.
Main Summary Screen	Specifies the main form to be launched on clicking on a record from Summary Result. This is applicable only for Dashboard screens.
Summary Web Services Required	Specifies whether a web service for the summary screen is required. **SUMMARYQUERY** action in the Actions screen has to be selected as well for enabling web services for the summary screen.
Data Block Fields	All the block fields present in the summary data block will appear in the text area to the left. The developer can select the block fields required in the summary screen and move them to the right. • **Field Name**: The required fields in the summary grid result can be moved to the right from the data block fields. • **Query Field**: Check the **Query** for fields that need to appear in the Query portion of the **Summary** screen. Note that the maximum number of Query fields for any screen is 12. • **LOV Name**: If the query option is checked, the properties button will be enabled, then on click of the button below screen will launch where LOV name and its corresponding return and bind variables can be defined, even **Min Search Char Length** details can also be mentioned here.

Table 36-1 (Cont.) Summary Details

Field	Description
Fields Ordering	The order of the fields in the summary screen can be rearranged. Here all the query fields will be shown to the developer in one table while all the fields in the result set will be shown in another table. The order of fields in both tables can be changed by the developer as per requirement.
Custom Buttons	Buttons can be added to the summary screen using the **Custom Buttons** Tab of the summary screen of Oracle FLEXCUBE Enterprise Limits and Collateral Management Development Workbench, same is shown below: • **Number of Rows**: Adding buttons to the summary screen can be of any number, and buttons can be arranged in more than one row. The maximum number of rows is 5. • **Number of Buttons/Row**: The number of Buttons per row can be maintained. • **Field Name**: This refers to the button name which is placed in the summary screen. • **Label**: Label Code for the button. • **Function Name**: JavaScript function which would be invoked on clicking the particular button is maintained here. Normally, this function would be written in the release-specific JavaScript file.

Preview of the Summary screen with different ordering of query fields and result set fields are shown below.

37 Generation and Deployment of files

This topic provides an overview of the generation and deployment process of the files.

Once the screen development is over, files have to be generated. Use generate icon to generate files. Once the files are generated, they have to be deployed in the FLEXCUBE environment for testing.

Refer Development Workbench- Screen Development II for the detailed explanation. After deployment of generated units, function id can be launched from the FLEXCUBE console. Unit testing of the function id can be carried out and any custom code can be added for additional functionality.

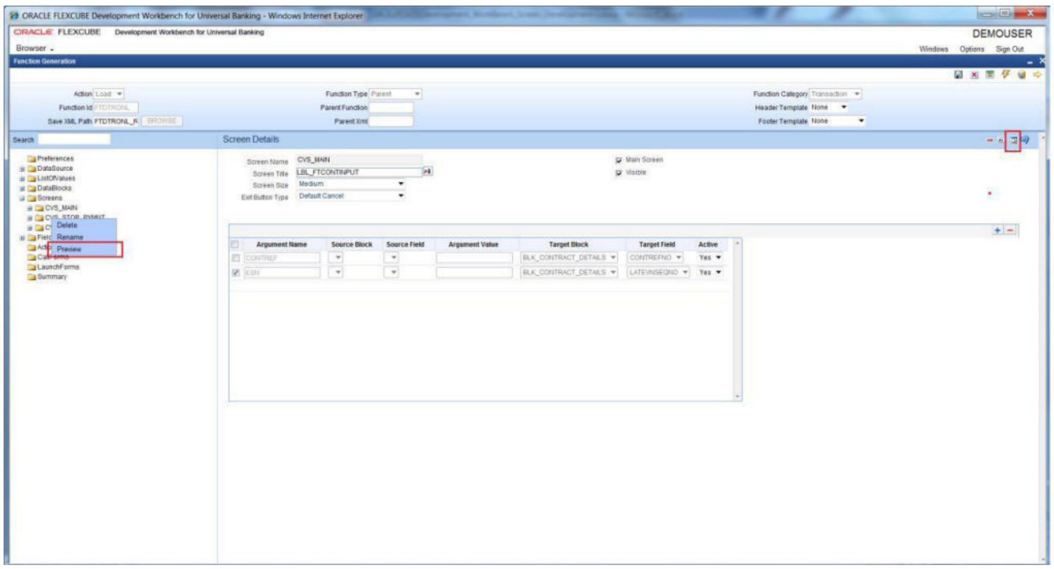
38 View Screen Preview

This topic provides systematic instructions to view preview of the screen.

Screen Preview gives the preview of the screen as it appears in the Oracle FLEXCUBE Universal Banking. It helps the developer to track the screen design changes much faster during the development phase.

1. Click on the **Preview** icon present in the top right portion of the screen.

Figure 38-1 Screen Details



2. Or Right-click on the particular **Screen** and select the **Preview** option from right-click menu.

> **Note:**
>
> By double-clicking on any field in the Preview, the developer can navigate directly to the particular data block field.

39
Locate Field

This topic provides systematic instructions to locate the field.

- Right-click on the **Data Block Field** and select **Locate Field** option from the right-click menu for locating a block field in the screen.

  Locate Field Feature helps determine the position of a field in a screen.

  Preview of the screen containing the field get launched and the position of the particular field get highlighted in the green background color.

40

Maintain Label Code

This topic provides systematic instructions to maintain label code.

Label Codes needs to be maintained in **CSTB_LABELS** (in the current environment language) for proper generation of files by ODT. Label Codes can be maintained in the FLEXCUBE through ODT itself.

1. Click on the **Label code** Icon found in the top right portion of the screen.

   Label Codes can be maintained through this screen. The screen provides the option to deploy the label codes directly to FLEXCUBE schema.

2. Maintain a new Label Code as follow:

   a. For maintaining a new label code, add a new label code as required in the**Label Code** field.

      Label code should start with **LBL_**.

   b. Provide the description of the field in the **Label Description** field.

   c. Select the **Label Type** based on the requirement.

   d. Select the **Operation** field as **Insert**.

   e. Click on the **Deploy** button.

      ODT will insert the label code provided to **CSTB_LABELS**. Language code will be taken from the language code of the current environment code. More than one label code can be deployed at one time. Note the following while creating new label codes:

      i. Label Code should start with **LBL_**.

      ii. Label Code should not contain any special characters except **Underscore(_)**.

3. Maintain missing labels as follow:

   a. Click on the **Fetch Missing Labels** button to fetch the missing labels for the screen.

      All the label codes present in the function id which is not maintained in **CSTB_LABELS** get fetched and displayed in the table.

   b. Provide a proper description if required and deploy the label codes.

4. Fetch all Label Codes for the screen as follow:

   a. Click on the **Fetch All Labels** button.

      All the label codes used in the function Id displayed in the table.

   b. If required, update the **Label Description** of any field.

5. Update an existing Label as follow:

   a. Fetch all Labels used in the function Id.

   b. Change the **Label Description** of the label codes for which update has to be done.

   c. Change the **Operation** to **Update**.

   d. Check the particular label checkbox and click on **Deploy**.

      The selected label codes get updated.

41 Search and Undo Feature

This topic provides an overview of the Search and Undo feature.

Search Feature

Search Feature allows the developer to locate an element in the tree easily. It is useful while browsing a function id with many nodes. Key In the string the user wants to find, ODT will search for the exact match of the string. If an exact match is not found, then the first character search will be done. The first match will be highlighted in green and the tree moves to the location. The next occurrence of the same string can be found by pressing Enter or Down Arrow key.

Undo Feature

Undo feature allows the developer to undo all the changes done in a particular screen of ODT. On clicking of **Undo** icon, the field values will be restored to the values before visiting the screen.