Development Workbench - Notifications Oracle Banking Trade Finance Release 14.8.0.0.0 Part No. G30072-01

[April] [2025]



Contents

1	Pref	ace 3
	1.1	Audience 3
2	Intro	oduction3
	How to	o use this Guide
3	Noti	fication – Getting started
	3.1	What is Notification
	3.2	Notification Trigger
4	Noti	fication Development
	4.1	Pre-request for Notification development and testing
	4.2	Notification specification
	4.3	Notification XML development
	4.4	Notification Process
	4.5	Development process in Development Workbench
	4.6	Notification Trigger
	4.7	Notifications
5	Dep	loy Notification
	5.1	Notification - Workbench related deployment 18
	5.2	Notification Trigger deployment
6	Test	Notification
	6.1	Notification flow
	6.2	Scheduler based notification 19
	6.3	MDB based notification flow
	6.4	Triggering notification and testing 21

1 Preface

This document describes the steps to develop the notification XML and notification trigger using Oracle FLEXCUBE Development Workbench for Universal Banking.

1.1 Audience

The Development Workbench Notification Development book is intended for the FLEXCUBE Application Developers who perform the following tasks:

Develop new Notification

To Use this manual, you need	conceptual and workir	ng knowledge of the below:
ro obe tino manual, you need	conceptual and worthin	ig into the uge of the below

Proficiency	Resources
FLEXCUBE UBS Development	FCUBS-FD01-01-01-Development
overview	Overview Guide
Interface Getting started	FCUBS-FD04-01-01-Interface Getting started
FLEXCUBE Development Workbench for Universal Banking Reference	User manuals
Web service development to have query web service in place	FCUBS-FD02-03-01-RAD Web Service Development

2 Introduction

How to use this Guide

The information in this guide includes:

- <u>Chapter 3, "Introduction"</u>
- <u>Chapter 4, "Notification Getting started"</u>
- <u>Chapter 5, "Notification Development "</u>
- <u>Chapter 6, "Deploy Notification"</u>
- <u>Chapter 7, "Test Notification"</u>

3 Notification - Getting started

3.1 What is Notification

Notification framework in FLEXCUBE UBS is used to communicate the business event happened in FLEXCUBE UBS to external systems. Depending upon the event, the XML message is pushed to external system's asynchronous Queues for their consumption.

3.2 Notification Trigger

Notification Triggers is developed to recognize the event and then invoke the notification process. This trigger is developed using Development Workbench.

4 Notification Development

4.1 Pre-request for Notification development and testing

Following are pre-request for notification development:

- Target FLEXCUBE Environment with Notification framework installed
- Development Workbench link mapped to the FLEXCUBE environment
- Required Query Web services developed and tested



Fig 5.1.1: Development of Notifications

4.2 Notification specification

Identify the notification requirement as below

- What is the Notification function ID name for RAD XML (Third character should be N)?
- What is the Notification code?
- What is the Base table in FLEXCUBE UBS that triggers the notification?
 - What operation at base table triggers (insert/update/delete)?
 - What is the where clause for filter?
- What is the query Web service to be used?
 - What is the operation?
 - What are the tags required?

Example;

0

- Notification function ID name *STNCUMOD*
- Notification code NOTIF_CA_CUSTACC_MOD
- Base table *STTM_CUST_ACCOUNT*
 - Operation *DELETE*
 - Filter Account class type in (S, U)
- Web service to be used *FCUBSAccService*
 - Operation *QueryCustAcc*
 - Request node *Cust-Account-IO*

4.3 Notification XML development

Notification RAD XML development creates the following files:

- RAD XML
- SPC
- SQL
- Static Data

4.4 Notification Process

There will be one trigger for the base table of notification and in case of multiple notifications sharing the same base table, there will be no new triggers created. Instead the same trigger created on the base table will be reused. This approach reduces the number of triggers being used for notifications.

4.5 Development process in Development Workbench

The notification development process in Workbench is split into two steps:

- 1. Notification Triggers
- 2. Notification Filter Procedure

The first step is to create notification triggers for base tables. The trigger generated from Workbench will be inserting key details into a static notification log table. The following details will be captured:

- Trigger code: A unique value to for a notification trigger.
- Base Table: The base table on which, the trigger is built.
- When Clause: A simple when clause for the notification trigger.

The second step is to capture details of notifications and generate the notification filter procedure. The following details are captured:

- Notification code: A unique value to identify a notification.
- Description: Meaningful description of the notification.
- Gateway Service:

4.6 Notification Trigger

After successful login to Development Workbench click on Notification Trigger option in the tree as shown below:



Fig 4.6.1: Notification trigger

Notification Trigger				– ×
				🎴 🛛 🗖 📝 🍃 🐬 🌒 🔿
				a
Trigger Code *		Base Table *		•
Description Firing Time	Before 🔻	PK Cols * PK Types *		
Each Record	Yes V	Data Types		
Selected Columns		Notification Codes		
Trigger When Clause	\sim			
Trigger Logic(Set \$N(
nigger Logie Octoria			*	
			Ŧ	

Fig 4.6.2: Notification trigger options

Notification Trigger we have two options - Add a new Trigger or Modify Existing one. **New:**

Notification Trigger		_ ×
		🎴 🛛 🖬 🌃 🎦 🖓 🎯 🔿
Trigger Code * TRG_CUSTADDR	Base Table * MSTM_CUST_ADDRESS	
Description Trigger for Customer Address Maintenand	PK Cols * CUSTOMER_NO~LOCATION~MEDIA	
Firing Time After -	PK Types * VARCHAR2~VARCHAR2~VARCHAR2	
Each Record Yes *	Data Types	
Selected Columns	Notification Codes	
Trigger When Clause (new.auth_stat='A')		
Trigger Logic(Set \$NOTIFY To Y/N)		
ELSE LOPERation := UPDATE; END IF;		



Trigger Code: A unique value to for a notification trigger. Follow the naming conversion it should start with **TRG_XXXX.**This is mandatory field. This attribute signifies the trigger code created as part of trigger creation step in OTD. Each notification will be linked to a trigger code.

Description : Information field. Meaningful description of Trigger is to be given.

Firing Time : Specify when trigger needs to fired. We can create only BEFORE and AFTER triggers for tables. (INSTEAD OF triggers are only available for views; typically they are used to implement view updates.) (After/Before).

Each Record: specify for each row required or not. If FOR EACH ROW option is specified, the trigger is row-level; otherwise, the trigger is statement-level. **(Yes/No)**

Base Table: The base table on which, the trigger is built. This is mandatory field. Select a valid table from available LOV next to the field.

Pk Cols: Enter Primary key fields of table in tilde (~) separated format. This is mandatory field.

Pk Types: Enter Primary key type of the corresponding primary key field. This is mandatory field.

Selected Columns and Data Types: Defunct

Trigger When Clause: A simple when clause for the notification trigger. A trigger restriction can be specified in the WHEN clause, enclosed by parentheses. The trigger restriction is a SQL condition that must be satisfied in order for Oracle to fire the trigger. This condition cannot contain sub queries. Without the WHEN clause, the trigger is fired for each row.

Notification Codes: If the trigger is associated with a specific notification code, then the particular notification code has to be provided in the field. If the trigger is shared across many Notifications, field can be left empty

Front-End Files	System Packages	Hook Paskages		hieta Data	Othern	^
RadXML Screen Xml System JS	Main Package Spac Main Package Body Notikitaton Trogers Upload Package Spac	Kernel Package Spac Kernel Package Body Ottelsfer Package Body Ottelsfer Package Body Ottelsfer Package Body Ottelsfer Package Body	Menu Datalis Datasource Details LOV Details Bicor: Datalis Conen Details Amendable Details Call form Details Summary Details	Latel Datalis Block PK Columns Function Call Forms Cataway Datalis Noticiation Datalis Function Parameters Purge Datalis	Xat5 Xat3With Annotations Screen Hinti Uptood Table Trigger Uptood Table Definition Archive Table Definition	
SLNo		File Name		File Type	Status	•
GWT	R#_TRG_OUSTADOR Irg		Т	RG		
e GWI	M_NOTIFICATION_TRIGGERSTRG_(USTADOR INC	11	WC		
TRG	_CUSTADOR_RAD sml		R	ADXML.		
	rror Description	Error Code	×			

Fig 4.6.4: Notification trigger: Generation

On successful save Notification Trigger will generate two files (gwtr#_<trg-code>.trg and GWTM_NOTIFICATION_TRIGGERS_<trg-code>.INC) user needs to compile them in FLEXCUBE schema.

Modify:

Notification Trigger Code * Trigger Code * Description Fiting Time Each Record Yes ~ Selected Columns Trigger When Clause Trigger When Clause Price * Data Types * Notification Codes Price * Price * Pri	5								_
Trigger Code * PE Base Table * Description PK Cols * Firing Time Before * Each Record PK Types * Selected Columns Notification Codes	Notification Trigger							-	×
Trigger Logic(Set SNOTIFY To Y/N)	Notification Trigger Trigger Code • [Description [Fring Time Each Record [Selected Columns [Trigger When Clause]	PK Cols • Before • Yes • Data Types Notification Codes	×	I	I	6	7		

Fig 4.6.5: Notification trigger: Modification

ication Trigger						a v	I I 🖌 🖕	
		_			_		223 223 223	
		Т	rigger Code	>	<			
Trigger Code *		15						
Description			Trigger Code					
Firing Time								
Each Record	Yes *							
Selected Columns				Search Reset				
Trigger When Clause				Search Reset				
nigiger vinen Giasse		pros		<< < 1nf5 > >>				
Trigger Logic(Sel SNC	OTIFY TO YAN (Trigger Code	^				
			CUST_AC_BRN_TFR					
			LOAN_BRN_TRFR					
			TD_AC_BRN_TFR					
			TRG_APP_DETAIL					
			TRG_BLKDELMSTR					
			TRG_BRTMMSTR					
			TRG_CATDET					
			TRG_CFRAMSTR					
			TRG_CHBK					
			TRG_CLAC					
			TRG_CLTMPRD					
			TRG_CONT					
			TRG_CSTBCTRT					
			TRG_CSTMPRD					
			TRG_CUST					
				Ψ.				

Fig 4.6.6: Notification trigger: Modification- Selecting Trigger name

Notification Trigger					- >
				×	l 📝 🍙 77 🎯 d
					Execute Query
Trigger Code * TRO	G_CUSTADDR				
Description		PK Cols *			
Firing Time Be	fore 👻	PK Types *			
Each Record Yes	S 🔻	Data Types			
Selected Columns		Notification Codes			
Trigger When Clause	\sim				
Trigger Logic(Set \$NOTIF)	Y 10 Y/N)				
			Ŧ		

Fig 4.6.7: Notification trigger: Modification- Entering values

🖬 🗵 🗊 🚰 🖗 🤤 d	Notification Trigger			- ×
Ingger Could in the control of the	Trigger Code Description Firing Time Each Record Selected Columns Trigger When Clause Trigger When Clause Trigger Under Code I Fi FWL LOP ELSE	After Yes Yes (new.auth_stat='A') (new.auth_stat='A') (OTIFY To YN) (old once_auth, N) ↔ Y THEN eration := "NSERT; eration := "NSERT; eration := "UPDATE;	PK Types * VARCHAR2-VARCHAR2-VARCHAR2 Data Types	



ation Trigger						3 7
erate Rad Files						
Front-End Files	System Packages	Hook Paokages		Meta Dala	Others	^
RadkiiL Borsen Xmi System JS	Main Package Spac Main Package Body Oblication Tragers Upload Package Spec	Kernel Package Spec Kernel Package Body Cluster Package Spec Cluster Package Body Custom Package Spec Custom Package Body	Menu Delalis Datasource Delalis LOV Octails Bioco Delalis Soreen Details Amendable Details Cail form Delalis Surmmary Cietails	Label Defails Block PK Columns Function Call Forms Caleway Defails Nohrication Defails Fundtion Parameters Purge Defails	Xada Xad With Annotaliona Screen Hinti Upioad Table Trigger Upioad Tables Definit Archive Table Definitio	
SLNo		File Name		File Type	Status	
	TRO_OUSTADORUIN		TR		Generalad +	
GWTM_M	OTIFICATION_TRIGGERS_TRO_C	USTADOR INC	INC	2	Generalad -	
TRG_CU	ISTADDR_RAD xml		RA	DXML	Generalad +	
nformation	Description	Error Code				
Request succes	sfully Processed	RD-SAVE-007				
		~ 			Genera	nte Exit

Fig 4.6.9: Notification trigger: Modification- Successful Generation

4.7 Notifications

Notifications Screen will be used to create new notification or modify existing notification; here we capture notification information for notification codes. We save notification details into xml.

	Notification Function		Action None -	Save XML	Path		
Notification Cod	e *		Mod	dule			
Descriptio	n		Module Descript	otion			
Notification Xs	d		Base Ta	able *			
Firing Tim	e 🔍 👻		PK C	Cols *			
Filter Typ	e 🗸 👻		PK Typ				
Gateway Servic	e			Full Screen Reply			
Gateway Operatio	n			HO Only			
Gateway IO Reques	st						
Type XSD Nam	e FY To Y/N & Refer Current Record as						
				*			
				~			
Web Service T	ags			*	+-		
Web Service T Order	ags Xsd Field	Table Field	Data Type	Maximum Length	* -		
		Table Field	Data Type	Maximum Length	+ -		

Fig 4.7.1: Notification Screen

Action: We can choose either new or Load action. New to create a new notification and Load is used to modify the existing one.

Save Xml Path: Specify the path to save notification xml. This would be considered only if the Save Mode is Client and Work Directory is specified as \$CURRENT_DIRECTORY

Notification Function: Specify the notification function-id name. *Conventions:*

Maximum 8 chars. 3rd letter must be 'N'. Example: FTNCONON

Notification Code: Enter the notification code to which we need to capture values. This is Mandatory field.

Development Workbench - Notifications

Recommended Convention for Notification Codes: NOTIF_<Module Code>_<Description> Example: NOTIF_LD_CONTRACT This is the notification indicating that a LD contract has been created/modified

Description: Information field. Meaningful description of the Notification has to be provided in the field

Module: This attribute signifies the module on which the notification is based.

Module Description: Information field. Module Description which would be defaulted from Module LOV

Notification XSD: Notification XSD name will have to be provided in the corresponding Field. Naming convention to be followed while naming Notification XSD is as follows

[Module Name] – [Notification Description] – Notif.xsd Example: FT-Contract-Notif.xsd

Notification XSD has to be provided only if no Gateway Web Service Query Operation is configured to the Notification

Base Table: Select the base table on which trigger needs to be applied.

Firing Time: Indicates the Operation on the base Table for which Notifications has to be sent. Options available are Insert, Update or Both

Filter Type: This attribute can take the following values.

- 1. Where clause
- 2. Plsql block

Pk Cols: Enter Primary key columns of the Base Table.

Pk Types: Enter Primary key field Data Types.

Provide details of Gateway Service, Operation, Type XSD Name and Full Screen Reply if a Query Web Service has to be mapped to the Notification

Gateway Operation: The gateway operation name to execute query for the mentioned Service.

Gateway Service: The gateway service to be used to get the full screen response.

Gateway IO Request: The gateway IO request node to be used in querying operation.

Development Workbench - Notifications

Type XSD Name: This field has to be entered if Notification is mapped to a Service and Request. Name of the Master Type XSD for the service and operation has to be provided here. This can be found in include portion of the Request Msg XSD of particular Service-Operation

Example: LC-Contract-Types.xsd

Full screen Reply: This attribute decides whether full screen or primary key notification response to be sent. This is applicable only if gateway Service details are provided

HO only: This attribute is used to send notification only from head office.

Filter Logic: The filter logic which decides whether the notification needs to be sent or not. This can be a simple where-clause on base table or a complex pl/sql block.

Web service Tags: The columns selected from base table as part of web service tags, will be used to send the full screen notification response. These tags defines the elements of Notification Xml when no Query service is mapped to it:

Front-End Files	System Packages	Hook Packages		Meta Data	Others
RadXML Screen Xml System JS	Main Package Spec Main Package Body Notification Triggers Upload Package Spec Upload Package Body	Kernel Package Body Kernel Package Body Cluster Package Body Cluster Package Body Custom Package Body Custom Package Body Custom Package Body	Menu Details Datasource Details LOV Details Block Details Screen Details Amendable Details Call form Details Summary Details	Label Details Block PK Columns Unuction Call Forms Gateway Details Volfication Details Function Parameters Purge Details	 Xsds Xsd With Annotations Screen Himi Upload Table Trigger Upload Tables Definition Archive Table Definition
SI.No		File Name	File Type		Status

Fig 4.7.2: Notification Screen generation

Front-End F	iles System Pack	ages	Hook Packages		Meta Gala			Others	
RadXML Screen Xml System JS	Main Package Spe Main Package Bod Notification Trigger Upload Package S Upload Package B	y s poc	Kemel Package Spec Kemel Package Body Cluster Package Body Cluster Package Body Cluster Package Body Custom Package Body Custom Package Body	Menu Delalis Diatasource Delalis LOV Details Bitock Delalis Screen Details Arriendable Details Cail form Delalis Summary Details	Label Delai Block PK Cr Function Ca Galeway De Vi Notification Function Pa Purge Delai	olumns Il Forms Talls Details rameters		ml	
SLNo			File Name		File Type		S	atus	
	flples_Itnconon_main.spc				BPC			¥	
	flpks_ftnconon_kernel.spc				SPC			*	
	fiples_ftnconon_main.sqi In	formation				×		*	
	fipks_ftnconon_kernei.sqi							w	
	GWTM_NOTIFICATIONS_MAST		Error Description		Error Code			w	
	GWTM_NOTIFICATION_TAG_N	! Re	quest successfully Processed		RD-SAVE-007			*	
	FTNCONON_RAD xml							•	
					Ok			Generate	Exit

Fig 4.7.3: Notification Screen Generation Successful

Modifying an Existing Notification RADXML The process of modifying an existing Notification RADXML is illustrated in the images below

Noticedon Function Actin Lod * Lod Screnz Xile Upload File Image: Constraint of the state	dication Mainferrance			
Upload File Upload File Up	Notification Function	Action Load 👻	Load Screen Xml	
Upload File Browse Upload File Browse Upload File Browse FIE CONSC File to Upload Conscience File	Noblication Code *	Module		
Upload File Browse Upload File Browse RECORD Comparison Name Date modified Type Desktop CCSCFTDUP_RAD.xml 12/23/2012.948 AM XMLF Desktop FTCPR/FT_RAD.xml 12/23/2012.948 AM XMLF Desktop FTCPR/FT_RAD.xml 12/23/2012.948 AM XMLF FTCPR/FT_RAD.xml 12/23/2012.948 AM XMLF FTCPR/FT_RAD.xml 12/23/2012.948 AM XMLF FTCDCNTH_RAD.xml 12/23/2012.948 AM XMLF FTDCCNTH_RAD.xml 12/23/2012.948 AM XMLF FTDCNTH_RAD.xml 12/23/2012.948 AM XMLF F		Choose File to Upload		×
Upload File Browse RECORD File Browse RECORD File Browse File File name Fi		COST → MAIN → FT → RADXML	✓ 4y Search RADXML	٩
Record Places Name Date modified Type Record Places Recent Places CSCFTDUP_RAD.xml 12/23/2012.948 AM XML F Deskop Libraries FTCCOND_RAD.xml 12/23/2012.948 AM XML F FTCCONDUR_RAD.xml 12/23/2012.948 AM XML F FTCCONDUR_RAD.xml 12/23/2012.948 AM XML F FTDCONDUR_RAD.xml 12/23/2012.948 AM XML F FTDCXFRA_RAD.xml 12/23/2012.948 AM XML F FTDMCKCH_RAD.xml 12/23/2012.948 AM XML F FTDMTIOL_RAD.xml 12/23/2012.948 AM XML F FTDMTIOL_RAD.xml<		Organize 👻 New folder	ii • 📋 (0
File name: All Files (".")		Recent Places SCFTDUP_RAD.xml FTCCQCILM_RAD.xm FTCCRJFT_RAD.xml FTCCQTH_RAD.xml FTCCQTH_RAD.xml FTDCCQTH_RAD.xml FTDCCQTH_RAD.xml FTDCQTH_RAD.xml FTDCQTH_RAD.xml FTDCQTH_RAD.xml FTDCQTH_RAD.xml FTDCQTH_RAD.xml FTDCQTH_RAD.xml FTDCQTH_RAD.xml FTDQTRA_RAD.xml FTDQTTRA_RAD.xml FTDQTTRA_RAD.Xml	12/23/2012 948 AM XM 12/23/2012 948 AM XMI 12/23/2012 948 AM XMI	LF LF LF LF LF LF LF LF LF
	Web Service Tags	File name:		

Fig 4.7.4: Notification Screen Loading

										×	¥
			Notification Function FT	NCONON	Actio	n Load 👻	Save Xml Pa	th FTNCONON_RAD.xml	BROWSE		
	Notificat	ation Code *	NOTIF_FT_CONTRACT			Module	FT	*=			
		Description	This is the notification in			Module Description	Funds Transfer				
	Notific	ication Xsd				Base Table	CSTB_CONTRACT				
	Fit	Firing Time	Insert 👻			PK Cols	CONTRACT_REF_NO				
	F	Filter Type	PIsql Block 🔹			PK Types	* VARCHAR2				
	Gatewa		FCUBSFTService	×1			Full Screen Reply				
			QueryContract	×1			HO Only				
			Contract-Details-IO								
	Type XS	XSD Name									
	Logic <mark>(</mark> Se			ecord as \$CURRENT_RECORD ■ = FT THEN \$NOTIFY := Υ; ELS		END IF; RETURN TRUE;	*				
Filter L	Logic(Se	\$CURRENT	r_RECORD.module_code			END IF; RETURN TRUE;	*				
Filter L	Logic(Se		r_RECORD.module_code			END IF; RETURN TRUE;	*	+			
V	Web Se	SCURRENT	r_RECORD.module_code	9 = 'FT' THEN \$NOTIFY := Y', ELS Table Field	E SNOTIFY := 'N';	Data Type	The second secon	+			
V	Under the second	SCURRENT	r_RECORD.module_code	9 = 'FT' THEN \$NOTIFY := 'Y', ELS	E SNOTIFY := 'N';		-	+			
Filter L	Under the second	Service Tag	r_RECORD.module_code	9 = 'FT' THEN \$NOTIFY := Y', ELS Table Field	E SNOTIFY := 'N';	Data Type	-				

Fig 4.7.5: Notification Screen Loaded

										-	7	
										× 1	7 🙀	Ę
		Notification Function	INCONON		Actic	n Load 👻	Save Xml P	ath FTNCONON_RAD.xml	BROWSE)		
Notifica	ation Code *	NOTIF_FT_CONTRACT		1		Module	FT	1				
		This is the notification in				Module Description	Funds Transfer					
Notific	cation Xsd			j		Base Table ³	CSTB_CONTRACT					
F	iring Time	Insert 👻				PK Cols	CONTRACT_REF_NO					
F	Filter Type	Pisql Block 🔹				PK Types *	VARCHAR2					
Gatew	ay Service	FCUBSFTService		* =			Full Screen Reply					
Gateway	Operation	QueryContract		×E			HO Only					
Gateway IC	O Request	Contract-Details-IO]								
Туре Х	(SD Name]								
		To Y/N & Refer Current R										
		_RECORD.module_cod	e = 'FT' THEN \$NC	DTIFY := 'Y'; ELSE	\$NOTIFY := 'N';	END IF; RETURN TRUE;	*					
			e = FT THEN \$NC	DTIFY := Y; ELSE	\$NOTIFY := 'N';	END IF; RETURN TRUE;	×					
	ervice Tag	s	e = FT THEN \$NC	DTIFY := Y; ELSE	\$NOTIFY := 'N';	END IF; RETURN TRUE;	*	+-				
Crde	ervice Tag	s Xsd Field		Table Field		Data Type	Maximum Length	+-				
Orde	ervice Tag	s	e = FT THEN \$NC	Table Field	SNOTIFY := 'N';		Maximum Length 64	+				
Crde	ervice Tag	s Xsd Field		Table Field _REF_NO		Data Type		+				

Fig 4.7.6: Notification Screen Loaded and Modified

5 Deploy Notification

5.1 Notification - Workbench related deployment

Compile the following files in Target FLEXCUBE UBS Database schema

- Notification Main Package generated from ODT
- Hook Packages
- GWTM_NOTIFICATION_TAG_MAP___<Notification Function ID>_.INC
- GWTM_NOTIFICATIONS_MASTER____<Notification Function ID>_.INC

5.2 Notification Trigger deployment

Compile the following files in Target FLEXCUBE UBS Database schema

- GWTM_NOTIFICATION_TRIGGERS_TRIG_CONTRACT.INC
- GWTR#_TRIG_CONTRACT.TRG

6 Test Notification

This section explains the run time notification flow and testing steps.

6.1 Notification flow

The notification process occurs as two parts:

- 1. Oracle JOBs created using FCJ Scheduler framework that sends data required for notification to an internal JMS queue.
- 2. Gateway MBD that lists on internal JMS queue, that picks the notification XMLs and prepare full web service response and send to external system queues.

6.2 Scheduler based notification

The Notification Process in FLEXCUBE can be done using the jobs scheduler as follows:

The trigger generated from Workbench will be inserting key details into a static notification log (STTB_NOTIFICATION)

Once Job is triggered, a request is sent to EJB layer from job execution class and the notification log table will be polled for unprocessed records.

Each unprocessed record is locked.

The record is verified against the notification maintenance and checked whether notification is to be sent or not.

If notification is to be sent, pre notification message xml is built and it is sent to internal NOTIFY_QUEUE(JMS queue) configured in Gateway layer.

The job is then rescheduled to fire next time based on the previous execution.

Refer Gateway Installation documents on how to setup the Queues.

Flow Chart for Notification Flow in Scheduler



6.3 MDB based notification flow

Notification processes in MDB are as follows:

- 1. Notification MDB listens on the internal NOTIFY_QUEUE(JMS queue)
- 2. On any message received, the MDB identifies which schema to connect using the JNDI name being present as part of the message xml.
- 3. Gateway notification processing package is called from MDB to build notifications.
- 4. In MDB, the notifications built is processed and sent to the destination specified in corresponding notification.
- 5. In case of exception the transaction is rolled back.
- 6. If all notifications are successfully processed, transaction is committed.





6.4 Triggering notification and testing

Follow the below steps to test notification:

- Simulate a case where base table under goes data change.
- Check record populated at STTB_NOTIFICATION table
- Check Notification message GWTBS_NOTIFICATIONS_LOG.NOTIFICATION_MESSAGE



Development Workbench - Notifications

April] [2025] Version 14.8.0.0.0

Oracle Financial Services Software Limited Oracle Park Off Western Express Highway Goregaon (East) Mumbai, Maharashtra 400 063 India

Worldwide Inquiries: Phone: +91 22 6718 3000 Fax:+91 22 6718 3001 www.oracle.com/financialservices/

Copyright © 2025, Oracle and/or its affiliates. All rights reserved.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate failsafe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

This software or hardware and documentation may provide access to or information on content, products and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.