Oracle® Banking Liquidity Management

Configuration Guide

Release 14.7.0.0.0
F78653-01
November 2022

Oracle Banking Liquidity Management Configuration Guide, Release 14.7.0.0.0

F78653-01

Contents

Preface

Purpose

This guide quickly get acquainted with the many functions every day on a routine basis as part of the End of Day (EOD).

Audience

This guide is intended for Back Office Data Entry Clerk, Back Office Managers/ Officers, Product Managers, End of Day Operators, and Financial Controller users.

Acronyms and Abbreviations

The list of acronyms and abbreviations that you are likely to find in the guide are as follows:

Table 1 Acronyms

Abbreviation	Description
API	Application Programming Interface
EOD	End of Day

List of Topics

This guide is organized as follows:

Table 2 List of Topics

Topics	Description
EOD Configuration	This topic provides the information about the instructions to perform the EOD operations.
Job definition Naming Convention	This topic provides the information about Job definition Naming Convention.
Oracle Banking Liquidity Management Job	This topic provides the information about the Oracle Banking Liquidity Management Job.
Intraday Jobs	This topic provides the information about the Intraday Jobs.
Functional Activity Codes	This topic provides the information about the Functional Activity Codes.

Related Documents

The related documents are as follows:

- *Oracle Banking Common Core User Guide*
- *Oracle Banking Liquidity Management User Guide*

1
EOD Configuration

This topic provide information about the EOD Configuration process.

This topic contains the following subtopics:

- Mapping Functional Activity Code
  The topic describes the information to map the functional activity code to perform EOD operations.
- Upload DSL
  This topic describes the systematic instructions to upload DSL in Business Process maintenance.
- Configure EOD
  This topic describes the systematic instructions to configure EOD operations
- Run EOD for branch
  This topic describes the systematic instructions to run the EOD for a branch.

1.1 Mapping Functional Activity Code

The topic describes the information to map the functional activity code to perform EOD operations.

The following functional activity code needs to be maintained in user's role to perform EOD operations:

CMC_FA_BRANCH_EOD_PROCESS

> **Note:**
>
> Refer to **Oracle Banking Security Management System User Guide** for the procedure to map the functional activity code in user's role.

1.2 Upload DSL

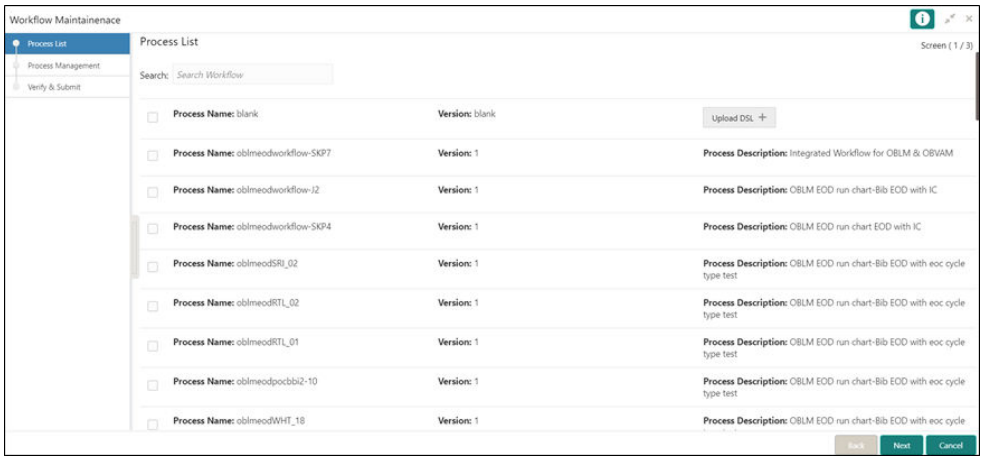
This topic describes the systematic instructions to upload DSL in Business Process maintenance.

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

1. Download the **OBLMEOD.json** file. This is a standard batch process definition script for Oracle Banking Liquidity Management that includes the list of batch tasks to be automatically executed in a sequence.
2. On **Home** Screen, under **Tasks** menu, click **Business Process Maintenance** to import, create or modify batch process definition

   The **Product List** screen displays.

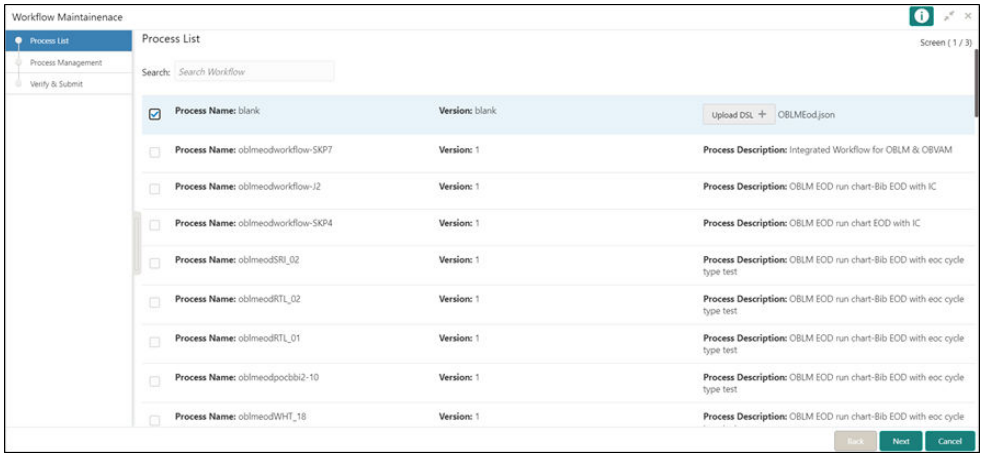
Figure 1-1 Process List



3. Select the **Process Name: blank** checkbox.
4. Click the **Upload DSL+** button to upload batch process definition.
5. Select the file **OBLMEod.json** from the local folder.

   The **Process List – Upload DSL** screen displays

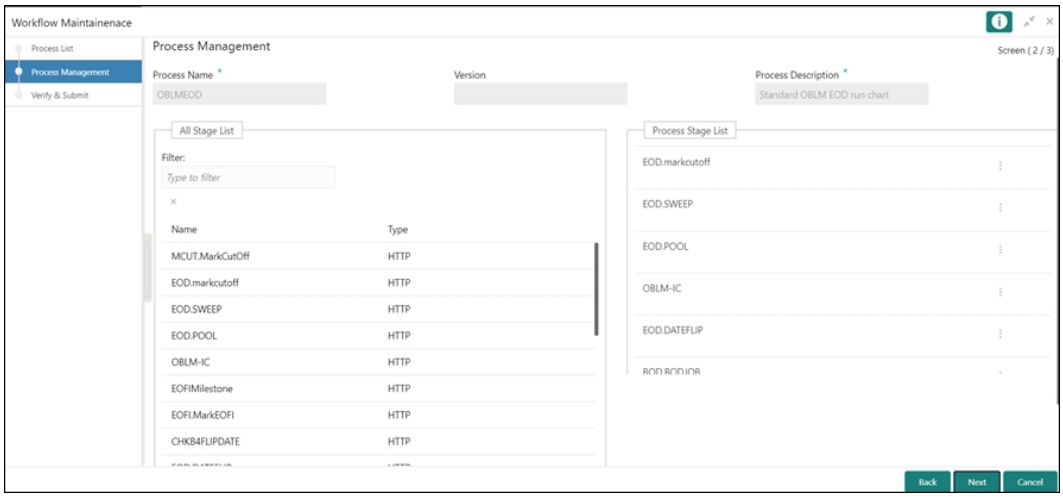
Figure 1-2 Process List – Upload DSL



6. Click **Next** button.

   The **Product Management** screen displays.

Figure 1-3 Process Management



7. Click **Next** button.

   The **Verify and Submit** screen displays.

8. Click **Review** or **Create Process** to register the batch.

1.3 Configure EOD

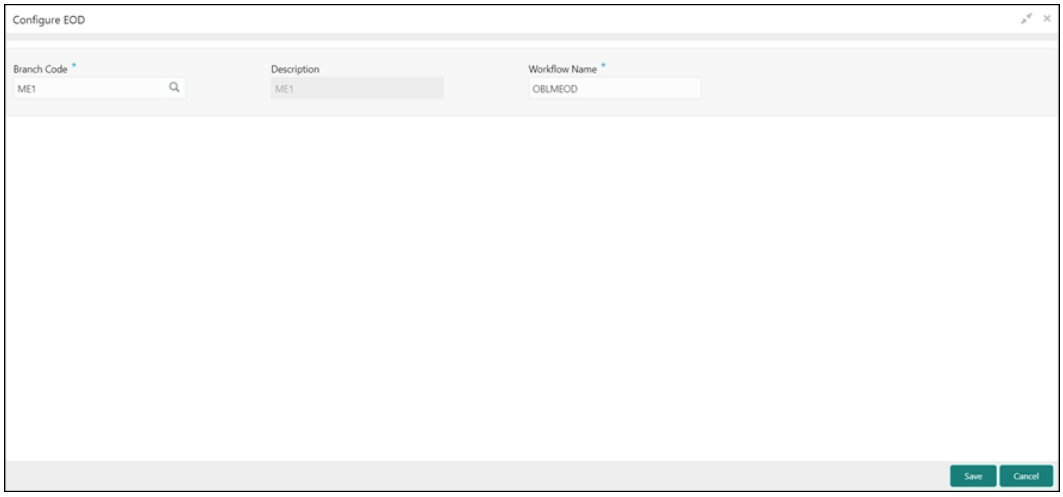
This topic describes the systematic instructions to configure EOD operations

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

1. On **Core Maintenance** menu, under **Branch EOD**, click **Configure EOD**.

   The **Configure EOD** screen displays.

Figure 1-4 Configure EOD



> **Note:**
>
> To configure batch for a branch, refer the **Configure Branch EOD** section in *Oracle Banking Common Core User Guide*.

2. Click **Search** icon to view and select the **Branch Code** to configure the batch.

> **Note:**
>
> The value specified in **Workflow name** field must be same as the **workflow name** attribute specified in 3rd line of batch script **OBLMEOD.json** file.

1.4 Run EOD for branch

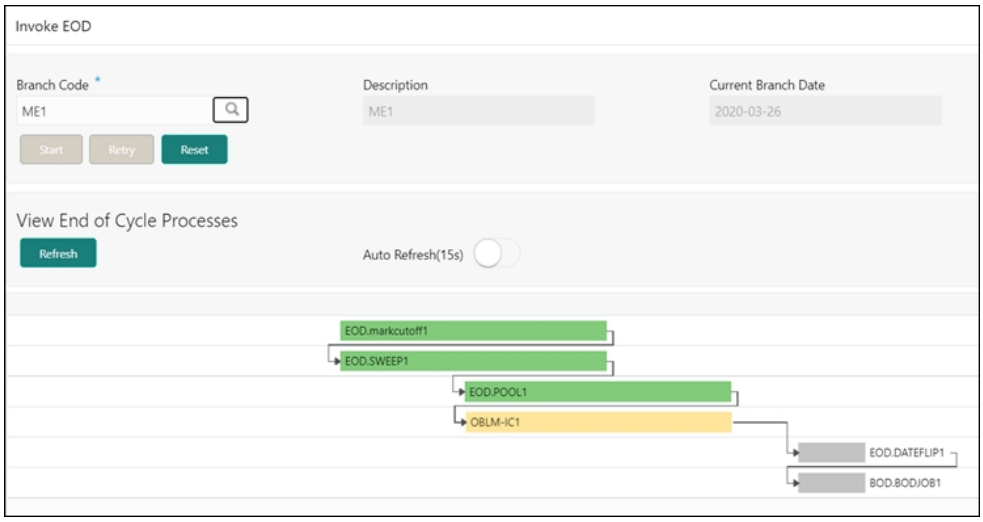
This topic describes the systematic instructions to run the EOD for a branch.

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

1. On **Home** Screen, click **Core Maintenance**. Under **Core Maintenance** menu, click **Branch EOD**.
2. Under **Branch EOD**, click **Invoke EOD**.

   The **Invoke EOD** screen displays.

Figure 1-5 Invoke EOD



3. Click **Search** icon to view and select the branch code to run EOD.
4. Click **Refresh** to view the current status of the branch.

2 Job Definition Naming Convention

This topic describes the naming convention that to be followed when a custom job is introduced as a task into EOD process.

1. **Milestone** task **name** must be prefixed with "**MS-**". Ex: MS-BranchCutOff

Milestone stage

Milestone stage will pause the batch execution till it is manually resumed.

Sample template for milestone stage

```
{
     "name": "MS-CHKAFTEREOTI",
     "taskReferenceName": "MS-CHKAFTEREOTI",
     "inputParameters": {
         "http_request": {
             "connectionTimeOut": "0",
             "readTimeOut": "0",
             "vipAddress": "CMC-BRANCH-SERVICES",
             "uri": "/cmc-branch-services/brancheod/milestone",
             "method": "POST",
             "headers": {
                 "appId": "CMNCORE",
                 "branchCode": "${workflow.input.branchCode}",
                 "userId": "${workflow.input.userID}"
             },
             "body": {
                 "data": [
                     {
                         "workflowId": "${workflow.workflowId}",
                         "taskId": "${CPEWF_TASK_ID}",
                         "waitTime": "5000"
                     }
                 ]
             }
         }
     },
     "type": "HTTP",
     "startDelay": 0,
     "optional": false,
     "asyncComplete": true
}
```

Steps to integrate Custom Jobs

1. If the custom job uses Oracle Banking Microservices Architecture Batch service, then use the below template to include the job as a task in EOD Flow definition.

```
{
   "type":"HTTP",
   "name":"<MilestoneCode.JobName>",
   "taskReferenceName":"<MilestoneCode.JobName>",
   "inputParameters":{
      "http_request":{
         "connectionTimeOut":"0",
         "readTimeOut":"0",
         "vipAddress":"PLATO-BATCH-SERVER",
         "uri":"/plato-batch-server/jobLauncher/launch/",
         "method":"POST",
         "headers":{
            "appId":"${workflow.input.appId}",
            "branchCode":"${workflow.input.branchCode}",
            "userId":"${workflow.input.userId}"
         },
         "body":{
            "jobName":"<JobName>",
            "jobParameters":[
               {
                  "key":"appId",
                  "value":"<Application ID of microservice>"
               },
               {
                  "key":"microServiceName",
                  "value":"<Microservice name>"
               },
               {
                  "key":"contextRoot",
                  "value":"<Context root of microservice>"
               },
               {
                  "key":"workflowId",
                  "value":"${workflow.workflowId}"
               },
               {
                  "key":"referenceTaskName",
                  "value":"<MilestoneCode.JobName>"
               },
               {
                  "key":"userId",
                  "value":"${workflow.input.userId}"
               },
               {
                  "key":"branchCode",
                  "value":"${workflow.input.branchCode}"
               },
               {
                  "key":"isCallback",
                  "value":"Y"
               },
               {
```

```
                    "key":"callbackType",
                    "value":"PLATOORCH"
                }
            ]
        }
    },
    "asyncComplete":true
},
"startDelay":0,
"optional":false,
"asyncComplete":true
}
```

2. If the custom job doesn't use the Oracle Banking Microservices Architecture Batch service. The Batch API is implemented as a synchronous call, use the below template to include the job as a task in EOD Flow definition.

```
{
   "type":"HTTP",
   "name":"<MilestoneCode.JobName>",
   "taskReferenceName":"<MilestoneCode.JobName>",
   "inputParameters":{
      "http_request":{
         "connectionTimeOut":"0",
         "readTimeOut":"0",
         "vipAddress":"<Microservice name registered in eureka>",
         "uri":"<relative URL>",
         "method":"<HTTP Method>",
         "headers":{
            "appId":"${workflow.input.appId}",
            "branchCode":"${workflow.input.branchCode}",
            "userId":"${workflow.input.userId}"
         }
      },
      "asyncComplete":false
   },
   "startDelay":0,
   "optional":false,
   "asyncComplete":true
}
```

> **Note:**
>
> HTTP Method - One of the GET, PUT, POST, DELETE, OPTIONS, HEAD

3. If the custom job doesn't uses Oracle Banking Microservice Architecture Batch service and if the Batch API is implemented as an asynchronous call, then call back needs to be implemented in the respective API. Use the below template to include the job as a task in EOD Flow Definition.

```
{
   "type":"HTTP",
   "name":"<MilestoneCode.JobName>",
```

```
    "taskReferenceName":"<MilestoneCode.JobName>",
    "inputParameters":{
       "http_request":{
          "connectionTimeOut":"0",
          "readTimeOut":"0",
          "vipAddress":"<Microservice name registered in eureka>",
          "uri":"<relative URL>",
          "method":"<HTTP Method>",
          "headers":{
             "appId":"${workflow.input.appId}",
             "branchCode":"${workflow.input.branchCode}",
             "userId":"${workflow.input.userId}"
          }
       },
       "asyncComplete":true
    },
    "startDelay":0,
    "optional":false,
    "asyncComplete":true
 }
```

Table 2-1 Batch API

Method	Post	Description
URL	http://<hostname>:<port>/plato-orch-service/api/tasks	-
Headers	userId : <Logged in user id> branchCode : <Logged in branch code> appId : platoorch Content-Type : application/json Accept : application/json	userId – User who updates the task branchCode – Branch where the update is performed.
Body	`{ "workflowInstanceId": "<EOD_Workflow_ID", "taskId": "<Task_ID>", "status": "<Status>"}`	EOD_Workflow_ID – A Workflow ID gets generated when EOD is invoked Task_ID – Unique task ID gets generated for each task once it starts Status – COMPLETED / FAILED_WITH_TERMINAL_ERROR / FAILED / IN_PROGRESS

Note:

asyncComplete – field in EOD workflow definition should be set to true if the Http task makes an asynchronous call and the task has to be updated explicitly by calling above update APIs. Only after successful update, next task will get executed.

3
Oracle Banking Liquidity Management Job

The topic describes the Oracle Banking Liquidity Management Job names and its descriptions.

Table 3-1 Oracle Banking Liquidity Management Job

S.No	EOD stage	Job Name	Description	Input Parameters
1	MCUT	markcutoff	Job will check for pending tasks and any existing running process before starting EOD	BranchCode
2	EOD	CHKPENDINGMAINT	Job will check pending maintenances that required approval.	BranchCode
3	EOD	SWEEP	Job will execute sweep process scheduled to run during EOD	BranchCode
4	EOD	POOL	Job will execute all pool structures.	BranchCode
5	EOD	PREIC	Job will execute the tasks that are required to run before starting Interest batch	BranchCode
6	EOD	IC.MARKCUTOFF	Job will check for pending tasks and any existing running process before starting Interest batch	BranchCode
7	EOD	OBLM-IC	Job will execute Interest batch	BranchCode
8	MS-EOFI	MS-EOFI	Milestone for the end of financial input	BranchCode
9	EOFI	MARKEOFI	Job will mark the end of financial input	BranchCode
10	MS-CHKB4FLIPDATE	MS-CHKB4FLIPDATE	Milestone for date flip	BranchCode
11	EOD	CMC.DATEFLIP	Job will change system date to next working date in common core	BranchCode
12	EOD	OBLM.DATEFLIP	Job will change system date to next working date in Oracle Banking Liquidity Management	BranchCode
13	EOD	RCUT.RELEASECUTOFF	Job will mark release cutoff after EOD.	BranchCode
14	EOD	IC.RELEASECUTOFF	Job will mark release cutoff for IC Batch.	BranchCode
15	BOD	BOD.REALLOC	Job will execute reallocation.	BranchCode
16	BOD	BOD.SWEEP	Job will execute the Reverse sweep and BOD sweeps in sequence	BranchCode
17	BOD	TI.MARKTI	Job will mark the transaction inputs	BranchCode

Table 3-1 (Cont.) Oracle Banking Liquidity Management Job

S.No	EOD stage	Job Name	Description	Input Parameters
18	BOD	BOD.ICL	Job will execute the ICL	BranchCode

4
Intraday Jobs

This topic provide information about the Intraday Jobs.

This topic contains the following subtopics:

- Create Task
  This topic describes the systematic instructions to create the task.
- Configure Tasks
  This topic describes the systematic instructions to configure the tasks.

4.1 Create Task

This topic describes the systematic instructions to create the task.

Oracle Banking Liquidity Management Intraday jobs required the following tasks to be created :

1. On **Home** screen, under **Task Management** menu, click **Create Task**.

   The **Create Task** screen displays.

Figure 4-1 Create Task

> **Note:**
>
> The fields, which are marked with an asterisk, are mandatory.

2. Specify the values mentioned in the following table.

Table 4-1 Intraday Job - Task Values

Sl. no	Task Name	Task Definition
1	OBLM_intraDayAccountPairSweepJob_INT_001	appId:::LMS;microServiceName:::oblm-sweep-services;contextRoot:::oblm-sweep-services;type:::schedule;jobName:::intraDayAccountPairSweepJob;cronExpression:::0 0/5 * * * ?;
2	OBLM_intraDayStructureSweepJob_INT_002	appId:::LMS;microServiceName:::oblm-sweep-services;contextRoot:::oblm-sweep-services;type:::schedule;jobName:::intraDayStructureSweepJob;cronExpression:::0 0/5 * * * ?;
3	OBLM_processMTHoldMessagesJob_INT_003	appId:::LMG;microServiceName:::oblm-messaging-services;contextRoot:::oblm-messaging-services;type:::schedule;jobName:::processMTHoldMessagesJob;cronExpression:::0 0/5 * * * ?;
4	OBLM_pendingPaymentsJob_INT_004	appId:::LMX;microServiceName:::oblm-integration-services;contextRoot:::oblm-integration-services;type:::schedule;jobName::: pendingPaymentsJob;cronExpression:::0 0/5 * * * ?;
5	OBLM_publishEventsLogJob_INT_005	appId:::LMX;microServiceName:::oblm-integration-services;contextRoot:::oblm-integration-services;type:::schedule;jobName:::publishEventsLogJob;cronExpression:::0 0/10 * * * ?;
6	OBLM_pendingReallocationJob_INT_006	appId:::LMX;microServiceName:::oblm-integration-services;contextRoot:::oblm-integration-services;type:::schedule;jobName:::pendingReallocationJob;cronExpression:::0 0/10 * * * ?;

3. Click **Create** to create the task for each Intraday job.

4.2 Configure Tasks

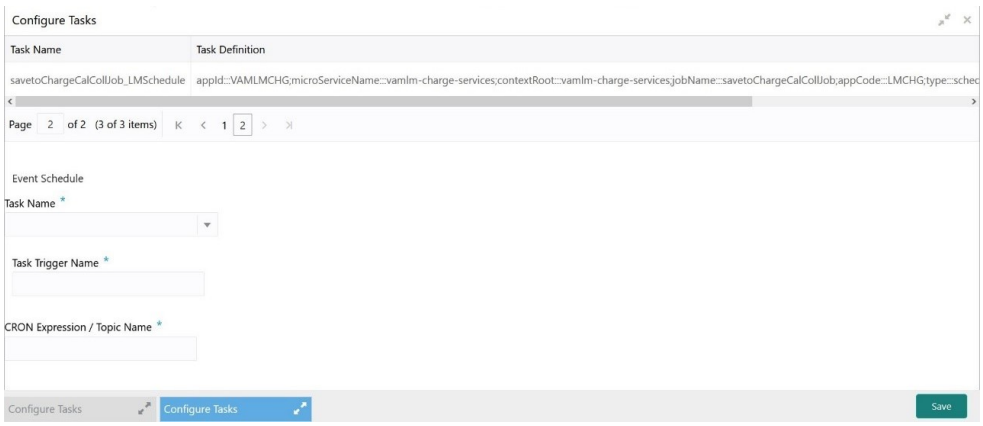
This topic describes the systematic instructions to configure the tasks.

The Configured intra-day jobs will get triggered as per the specified Cron Expression, for the Create Task the scheduler needs to be configured as shown as follows.

1. On **Home** screen, under **Task Management** menu, click **Configure Tasks**.

   The **Configure Tasks** screen displays.

Figure 4-2 Configure Tasks



> **Note:**
>
> The fields, which are marked with an asterisk, are mandatory.

2. Select the **Schedule** button.
3. Select the task name from the **Task Name** drop-down list.
4. Specify the trigger name in **Task Trigger Name** field.
5. Specify the required CRON expression in **CRON Expression** field.
6. Click **Save** to configure the task.

A
Functional Activity Codes

Table A-1 List of Functional Activity Codes

Functional Activity Code	Purpose
LMS_FA_SWEEPDATA_VIEW	This functional activity code is used to fetch the sweep data to provide the next execution date in case of Intraday account pair sweeps and to fetch account pairs based on frequency in case of EOD/BOD account pair executions
LMS_FA_SWEEPDATA_CREATE	This functional activity code is used to create the sweep data during structure creation
LMS_FA_SWEEPDATA_UPDATE	This functional activity code is used to update existing sweep data during structure modification
LMX_FA_PENDING_AUTH_VIEW	This functional activity code is used to view the maintenances pending for authorization
LMX_FA_HAS_PENDING_AUTH	This functional activity code is used to check whether the branch has any pending maintenances for authorization

Index