# 

**PRODUCT - Country Model Bank Development & Client Specific Development**

**TECHNICAL SPECIFICATION DOCUMENT**

**Husbanken**

**TSD\_LM04 - Invoices and credit notes- Loan Maintenance**

Document History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version | Date | Author / Changed By | Status | Change Description |
| 0.1 | 01/06/2021 | Marimuthu M | New |  |
| 1.0 | 28/10/2021 | Vinod | Baselined |  |
| 1.1 | 28/03/2023 | Marimuthu M | Updated | Changed routine logic of 8.12 HusRepaymentApi to solve EHL-1706 and based on latest FSD |
| 1.2 | 15/06/2023 | Marimuthu M | Updated | Update for APR solution |
| 1.3 | 17/07/2023 | Marimuthu M | Updated | Update for Record cost FSD (EHL-890) |
| 2.0 | 01/09/2023 | Umar | Baselined | Temenos review |
| 2.1 | 19/10/2023 | Marimuthu M | Updated | Update for EHL-1571 Payment Subtype |
| 2.2 | 22/11/2023 | Marimuthu M | Updated | Update for EHL-822 Validation to handle instalments frequency changes |
| 2.3 | 30/11/2023 | Lakshmipriya J | Reviewed | Review comments updated |
| 2.4 | 11/12/2023 | Marimuthu M | Updated | Closed review comments |

Sign Off History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Temenos Authorisation | Client Authorisation |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[1. Management Summary 4](#_Toc79600126)

[2. Functional Overview 5](#_Toc79600127)

[3. Operational Overview 5](#_Toc79600128)

[4. Reusable Components 5](#_Toc79600129)

[5. New / Amended Files 6](#_Toc79600130)

[6. Technical Overview 15](#_Toc79600131)

[7. New Tables 15](#_Toc79600132)

[8. Subroutines 28](#_Toc79600133)

[9. Appendix 58](#_Toc79600134)

[10. Accounting 58](#_Toc79600135)

[11. Limits 58](#_Toc79600136)

[12. COB Processing 58](#_Toc79600137)

[13. Assumptions 58](#_Toc79600138)

[14. Dependencies/Exclusions 58](#_Toc79600139)

[15. Testing Notes 59](#_Toc79600140)

[16. Packaging / Installation 59](#_Toc79600141)

[17. Data Migration 59](#_Toc79600142)

# Management Summary

## Background

T24 shall handle billing and collections. Billing is mainly the sending of instalment invoices for cases to the borrowers, but it can also happen when grants need to be paid off as the conditions are no longer met. The system shall also request payments to handle extraordinary repayments, early payoffs and the payment of premium for early termination of fixed-rate interest agreements. Husbanken allows postponements of instalment invoices. Husbanken uses the National Collection Agency (NCA), a subordinate agency of the Norwegian Tax Administration as external debt collectors.

## Modification Overview

Objective of this document is to provide solution to Invoices and credit notes for Husbanken.

## Reference Documents

|  |  |  |  |
| --- | --- | --- | --- |
| Document Name | Description | Author, Company | Embedded document / Reference |
| FSD\_LM04 - Invoices and credit notes - Loan Maintenance | FSD | Thomas Smits | [FSD Link](https://husbanken365.sharepoint.com/sites/NyttLaanesystem2-Prosjektdokumenteksternt/Shared%20Documents/Forms/AllItems.aspx?id=%2Fsites%2FNyttLaanesystem2%2DProsjektdokumenteksternt%2FShared%20Documents%2FProsjektdokument%20%28eksternt%29%2FExt%2010%20Engineering%2F30%20Design%2F20%20FSDs%2FFSD%2003%20%2D%20Approved&viewid=1d4df54d%2D2568%2D412d%2Db175%2D4bf268d303c1) |
| Requirement Tracebility Matrix | RTM | Susila/Marimuthu | [RTM Link](https://syncordisconsulting.sharepoint.com/sites/HusbankenOffshore/Documents%20partages/Forms/AllItems.aspx?FolderCTID=0x012000DE7F537B99E4D54687B81F5A0280C56B&isAscending=false&id=%2Fsites%2FHusbankenOffshore%2FDocuments%20partages%2FGeneral%2F02%2E%20RTM&sortField=Modified&viewid=3453ce9b%2Dcd4c%2D4d20%2Da5f7%2D1dee1690c974) |

### Reference Number of the Requirement (ODR) which is dependant on this Requirement

NA

## Glossary (Optional)

This section provides glossary & descriptions that are used in this document.

|  |  |
| --- | --- |
| Glossary | Description |
| BRD | Business Requirement Document |
| FSD | Functional Specification Document |
| UC | Use Case |
| API | Application Programming Interface |
| UXP Browser | User Interface Infinity software from Temenos |
| E-APP | E Application |

# Functional Overview

## Existing Functionality

No existing T24 functionality is used in this solution.

## New Functionality

1. The system initiates the invoices 20 working days before the payment due date.
2. When generating a payment request for Pay-Off, the banker should be able to combine multiple loans and grants in the Payment Request, and the payment from the customer should then be split towards the different arrangements
3. Invoicing functionality - Invoices need to be sent to the customers to request for repayments (any type: principal, interests, fees, early termination, grants…)
4. After receiving the signed amendment request, if the amendment results in compensation to be given to the customer, T24 needs to generate a credit note (either through Husbankens Document Service or through PEPPOL), and perform the compensation booking by Debiting a P&L category "Loss on Future Interest" and Crediting the Loan account resulting in a reduction of the Outstanding Capital.
5. Instalment fee - Amount depends on the invoicing channel. A logic needs to be created to charge automatically the correct fee amount.
6. A single instalment fee needs to be charged per invoice. If the invoice combines payments from several loans, there should be a logic to trigger the fee from the loan with the greatest maturity date.

# Operational Overview

NA

# Reusable Components

## Pack Reusability

### Reusable – (Yes/No)

### Reason

NA

## Reusable Components Table

NA

# New / Amended Files

## Local Table

### Local Table Name

NA

## LOCAL.REF.TABLE

NA

## Enquiry

### HUS.INVOICE.RESEND

This enquiry will be used to search invoice and resend it.

|  |  |
| --- | --- |
| FIELD.NAME | VALUE |
| PAGE.SIZE | 4,40 |
| FILE.NAME | EB.HB.INVOICE.STAGING |
| FIXED.SELECTION |  |
| SELECTION.FLDS.1 | @ID |
| SEL.LABEL.1 | Kid Number |
| REQUIRED.SEL.1 |  |
| SELECTION.FLDS.2 |  |
| SEL.LABEL.2 | CASE.ID |
| REQUIRED.SEL.2 | Case Id |

**Field Definitions:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Label in Output** | **Field Name** | **Operation** | **Column** | **Conversion** | **Other Attributes** |
| Kid Number | KID.NUMBER | @ID | 1 |  |  |
| Customer ID | CUSTOMER.ID | CUSTOMER.ID | 2 |  | Multi-Value field |
| Register ID | REGISTER.ID | REGISTER.ID | 3 |  |  |
| Case Id | CASE.ID | CASE.ID | 4 |  |  |
| Issue Date | ISSUE.DATE | ISSUE.DATE | 5 |  |  |
| Due Date | DUE.DATE | DUE.DATE | 6 |  |  |
|  | ZERORECORDS | "No Invoice found" | 1 |  | Display Break = NONE  Field Display Type = Class-enq Norecs  Section = Header |

**Drilldown and other special attributes:**

|  |  |
| --- | --- |
| Enquiry Name | EB.HB.INVOICE.STAGING,HUS.RESEND I KID.NUMBER |
| Label Field | KID.NUMBER |
| Nxt Desc | Resend Invoice[edit.gif |
| Description | Invoice Resend |
| Short Description | Invoice Resend |
| Attributes | Zerorecorddisplay |

### HUS.INTERFACE.RECALL

This enquiry will be used to recall interface.

|  |  |
| --- | --- |
| FIELD.NAME | VALUE |
| PAGE.SIZE | 4,40 |
| FILE.NAME | EB.HB.INVOICE.DETAILS |
| FIXED.SELECTION |  |
| BUILD.ROUTINE | HusBulIntRecall |
| SELECTION.FLDS.1 | STATUS |
| SEL.LABEL.1 | Status |
| REQUIRED.SEL.1 |  |
| SELECTION.FLDS.2 |  |
| SEL.LABEL.2 | DD.STATUS |
| REQUIRED.SEL.2 | DD Status |

**Field Definitions:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Label in Output** | **Field Name** | **Operation** | **Column** | **Conversion** | **Other Attributes** |
| Invoice Detail ID | INV.DET.ID | @ID | 1 |  |  |
| Type | TYPE |  | 2 |  |  |
| Invoice Status | STATUS |  | 3 |  |  |
| Narrative | STATUS.NARR |  | 4 |  |  |
| DD Status | DD.STATUS |  | 5 |  |  |
| DD Narrative | DD.STATUS.NARR |  | 6 |  |  |
|  | ZERORECORDS | "No failure records found" | 1 |  | Display Break = NONE  Field Display Type = Class-enq Norecs  Section = Header |

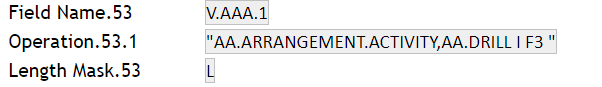
**Drilldown and other special attributes:**

|  |  |
| --- | --- |
| Enquiry Name | EB.HB.INVOICE.DETAILS,HUS.RECALL S INV.DET.ID |
| Label Field | INV.DET.ID |
| Nxt Desc | View[view.gif |
| Enquiry Name | EB.HB.INVOICE.DETAILS,HUS.RECALL I INV.DET.ID |
| Label Field | INV.DET.ID |
| Nxt Desc | Recall Interface[edit.gif |
| Description | Interface Recall |
| Short Description | Interface Recall |
| Attributes | Zerorecorddisplay |

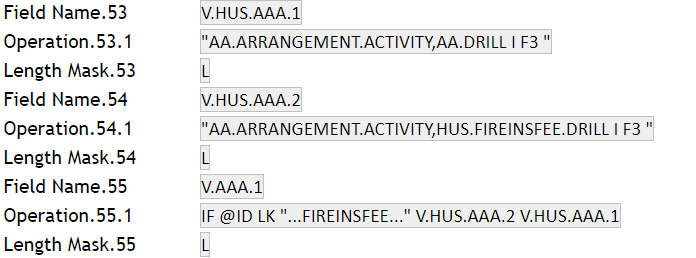
### HUS.AA.DETAILS.NEW.ACTIVITIES

Copy core enquiry AA.DETAILS.NEW.ACTIVITIES and create new enquiry with name HUS.AA.DETAILS.NEW.ACTIVITIES and do the below changes.

Existing value in core enquiry:



Replace 53rd multi-value with below values and add two new multi-value as shown below.



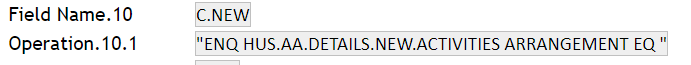
### HUS.AA.OVERVIEW-HEADING.LIV

Copy core enquiry AA.OVERVIEW-HEADING.LIV and create new enquiry with name HUS.AA.OVERVIEW-HEADING.LIV and do the below changes.

Existing value in core enquiry:



Replace 10.1 multi-value with new value as shown below



## Context Enquiry

### [Context Enquiry Name]

NA

## Version

### EB.HB.INVOICE.DETAILS,HUS.OFS

This is vanilla version and it will be used to post OFS message to update EB.HB.INVOICE.DETAILS record

NO.OF.AUTH = 0

EXC.INC.RTN = Yes

### EB.HB.INVOICE.STAGING,HUS.RESEND

This version will be used to resend invoice by setting RESEND flag.

All the fields in EB.HB.INVOICE.STAGING should be Noinput field except RESEND and RESEND field should be in first row of the version.

NO.OF.AUTH = 0

### EB.HB.INVOICE.STAGING,HUS.OFS

This is vanilla version and it will be used to post OFS message to update EB.HB.INVOICE.STAGING record

NO.OF.AUTH = 0

### EB.HB.INVOICE.DETAILS,HUS.INPUT

This is vanilla version and it will be used to post OFS message to create new record in EB.HB.INVOICE.DETAILS record.

NO.OF.AUTH = 0

EXC.INC.RTN = Yes

### EB.HB.INVOICE.DETAILS,HUS.DD.CANCEL

This is vanilla version and it will be used to post OFS message to cancel invoices related to direct debit.

NO.OF.AUTH = 0

EXC.INC.RTN = Yes

### EB.HB.INVOICE.DETAILS,HUS.DUEDATE.CHANGE

All the fields in EB.HB.INVOICE.DETAILS should be Noinput field except DUE.DATE and DUE.DATE field should be in first row of the version.

NO.OF.AUTH = 0

EXC.INC.RTN = Yes

Auto Default: DD.STATUS = SENDING-CHANGE-DD-DUE-DATE

### EB.HB.INVOICE.LEFTOVER,HUS.OFS

This is vanilla version and it will be used to post OFS message to create/update record in EB.HB.INVOICE.LEFTOVER

NO.OF.AUTH = 0

### EB.HUS.LA.CASE,HUS.INPUT

This is vanilla version and it will be used by interface team to trigger IF event

NO.OF.AUTH = 0

### EB.HB.INVOICE.DETAILS,HUS.RECALL

This is vanilla version and it will be used in an enquiry as drilldown version to recall interface

NO.OF.AUTH = 0

EXC.INC.RTN = Yes

Auto New content Routine: HusAncStatusUpd

### EB.HB.INVOICE.DETAILS,INPUT

This is version will be used by middleware team to update EB.HB.INVOICE.DETAILS records

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| T24 Field Name | Version Field Name | Visible to the User Y/N | Mandatory | Other Attributes |
| TYPE | Type | Y |  | No-input Field |
| CURRENCY | Currency | Y |  | No-input Field |
| CUSTOMER.TYPE | Customer Type | Y |  | No-input Field |
| CUSTOMER.REF | Customer Reference | Y |  |  |
| CUSTOMER.ID-1 | Customer | Y |  | No-input Field |
| REGISTER.ID-1 | Register Id | Y |  | No-input Field |
| DBT.REST.STATUS-1 | Debt. Rest Status | Y |  | No-input Field |
| DUE.DATE | Due Date | Y |  |  |
| INITIAL.DUE.DATE | Initial Due Date | Y |  | No-input Field |
| TOTAL.AMOUNT | Total Amount | Y |  | No-input Field |
| CHANNEL | Invoice Channel | Y |  | No-input Field |
| PAYMENT.CHANNEL | Payment Channel | Y |  | No-input Field |
| DD.STATUS | DD Status | Y |  | No-input Field |
| DD.STATUS.NARR | DD Status Narrative | Y |  | No-input Field |
| SEND.PMT.ADVICE | SEND.PMT.ADVICE | Y |  |  |

|  |  |
| --- | --- |
| Associated Version | EB.HB.INVOICE.DETAILS,HUS.ARR.DET  EB.HB.INVOICE.DETAILS,HUS.PRE.DET  EB.HB.INVOICE.DETAILS,HUS.AUDIT |
| Number of authorization | 0 |
| Screen Title | Invoice Details |
| Description | Invoice Details |

**Layout:**



### EB.HB.INVOICE.DETAILS,HUS.ARR.DET

This version will be attached as associated version in EB.HB.INVOICE.DETAILS,INPUT

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| T24 Field Name | Version Field Name | Visible to the User Y/N | Mandatory | Other Attributes |
| ARRANGEMENT.ID-1 | Arrangement Id | Y |  | No-input Field |
| LINE.ID-1.1 | Line | Y |  | No-input Field |
| AMOUNT.TYPE-1.1 | Type | Y |  | No-input Field |
| AMOUNT-1.1 | Amount | Y |  | No-input Field |
| BILL.ID-1.1 | Bill ID | Y |  | No-input Field |

|  |  |
| --- | --- |
| Number of authorization | 0 |
| Screen Title | Arrangements |
| Description | Arrangements |

### EB.HB.INVOICE.DETAILS,HUS.PRE.DET

This version will be attached as associated version in EB.HB.INVOICE.DETAILS,INPUT

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| T24 Field Name | Version Field Name | Visible to the User Y/N | Mandatory | Other Attributes |
| ISSUE.DATE | Issue Date | Y |  | No-input Field |
| KID.NUMBER | Kid Number | Y |  | No-input Field |
| STATUS | Status | Y |  | No-input Field |
| STATUS.NARR | Status Narrative | Y |  | No-input Field |
| TYPE | Type | Y |  | No-input Field |
| DUE.DATE | Due Date | Y |  | No-input Field |
| PREV.ISSUE.DATE-1 | Prev. Issue Date | Y |  | No-input Field |
| PREV.KID.NUMBER-1 | Prev. KID Number | Y |  | No-input Field |
| PREV.STATUS-1 | Prev. Status | Y |  | No-input Field |
| PREV.STATUS.NARR-1 | Prev. Status Narrative | Y |  | No-input Field |
| PREV.TYPE-1 | Prev. Type | Y |  | No-input Field |
| PREV.DUE.DATE-1 | Prev. Due Date | Y |  | No-input Field |

|  |  |
| --- | --- |
| Number of authorization | 0 |
| Screen Title | Status |
| Description | Status |

### EB.HB.INVOICE.DETAILS,HUS.AUDIT

This version will be attached as associated version in EB.HB.INVOICE.DETAILS,INPUT

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| T24 Field Name | Version Field Name | Visible to the User Y/N | Mandatory | Other Attributes |
| OVERRIDE-1 | Override | Y |  |  |
| RECORD.STATUS | Record Status | Y |  |  |
| CURR.NO | Current Number | Y |  |  |
| INPUTTER-1 | Inputter | Y |  |  |
| AUTHORISER | Authoriser | Y |  |  |
| DATE.TIME-1 | Date Time | Y |  |  |
| CO.CODE | Company Code | Y |  |  |
| DEPT.CODE | Department Code | Y |  |  |
| AUDITOR.CODE | Auditor | Y |  |  |
| AUDIT.DATE.TIME | Audit Date Time | Y |  |  |

|  |  |
| --- | --- |
| Number of authorization | 0 |
| Screen Title | Audit |
| Description | Audit |

### AA.ARRANGEMENT.ACTIVITY,HUS.FIREINSFEE.DRILL

Copy core version AA.ARRANGEMENT.ACTIVITY,AA.DRILL and create version with name AA.ARRANGEMENT.ACTIVITY,HUS.FIREINSFEE.DRILL

NO.OF.AUTH = 0

### AA.SIMULATION.RUNNER,HUS.AA

Copy core version AA.SIMULATION.RUNNER,AA and create version with name AA.SIMULATION.RUNNER,HUS.AA

Additionally do the below changes:

* Default T.ACTIVITY field with the value “LENDING-APPLYPAYMENT-PR.PRINCIPAL.DECREASE”
* Default EXECUTE.SIMULATION field with the value “NO”
* Default T.RUN.ACT field with the value “YES”

### EB.HB.INVOICE.DETAILS,HUS.PAYOFF

Create vanilla version to post OFS message to create record in EB.HB.INVOICE.DETAILS and attach HusAuthPayOffTypeStaging routine as authorisation routine.

NO.OF.AUTH = 0

### EB.HB.INVOICE.STAGING,HUS.PAYOFF

Create vanilla version to post OFS message to create record in EB.HB.INVOICE.STAGING.

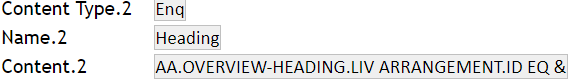
NO.OF.AUTH = 0

## Composite Screen

### HUS.AA.OVERVIEW.AL

Amend the existing composite screen and do the below changes

Existing value:



Replace Content.2 multi-value with new value as shown below.

A close up of a sign

Description automatically generated

Note: This composite screen developed as part of TSD\_BP04\_ Transfer of Loans

## Tabbed Screen

### [TABB Name]

NA

## HelpText Menu

### HUS.EXTRA.REPAY

|  |  |
| --- | --- |
| FIELD.NAME | VALUE |
| APPLICATION.1 | AA.SIMULATION.RUNNER,HUS.AA I F3 |
| DESCRIPT.1 | Simulate Extra-ordinary/Individual Repayment |

## I-DESCRIPTORS

NA

## STANDARD.SELECTION

NA

## Data Records

### EB.LOOKUP

Create record in EB.LOOKUP with below details

|  |  |
| --- | --- |
| Record id | Description |
| HB.IN.DET.STATUS\*NEW | During the day before the EHF file is transferred to HIS |
| HB.IN.DET.STATUS\*SENDING | EHF file transferred to HIS |
| HB.IN.DET.STATUS\*SENT | Successfully processed by HIS |
| HB.IN.DET.STATUS\*RECEIVED | Document has been received by the recipient |
| HB.IN.DET.STATUS\*OPENED | Document has been opened by the recipient |
| HB.IN.DET.STATUS\*FAILED | Failure in HIS |
| HB.IN.DET.STATUS\*CLOSED | The invoice is closed |
| HB.IN.DET.STATUS\*SENDING-GET-CHANNEL | Sending to get Channel |
| HB.IN.DET.STATUS\*FAILED-GET-CHANNEL | Get Channel Failed |
| HB.IN.TYPE\*CREDIT.NOTE | Credit Note |
| HB.IN.TYPE\*CREDIT.NOTE.SHARELOSS | Credit Note Share Loss |
| HB.IN.TYPE\*DEBT.COLLECTION.NOTICE | Debt Collection Notice |
| HB.IN.TYPE\*FINAL.DEMAND.PAYMENT | Final Demand Payment |
| HB.IN.TYPE\*INVOICE | Invoice |
| HB.IN.TYPE\*PAYMENT.REQUEST | Payment Request |
| HB.IN.TYPE\*PAYMENT.REQUEST.SHARELOSS | Payment Request Share Loss |
| HB.IN.TYPE\*PAYMENT.REQUEST.PAYOFF | Payment Request PayOff |
| HB.IN.TYPE\*PAYMENT.REQUEST.PAYOFFGRANT | Payment Request PayOff Grant |
| HB.IN.TYPE\*PAYMENT.REQUEST.REPAYGRANT | Payment Request RepayGrant |
| HB.IN.TYPE\*PAYMENT.REQUEST.PAYEXTRA | Payment Request PayExtra |
| HB.IN.TYPE\*PAYMENT.REQUEST.PAYINDEBT | Payment Request PayInDebt |
| HB.IN.TYPE\*PAYMENT.REQUEST.PREMIUM | Payment Request Premium |

### EB.ALTERNATE.KEY

Create record in EB.ALTERNATE.KEY with below details.

|  |  |
| --- | --- |
| Field Name | Field Description |
| @ID | EB.HB.INVOICE.DETAILS |
| Alt Key Max Length | 30 |
| Alt Key Field | KID.NUMBER |
| Concat Type | System |
| Access Method | Read |
| Unique | System |

### OFS.SOURCE

Create record in OFS.SOURCE with below details

|  |  |
| --- | --- |
| Field Name | Field Value |
| @ID | PAYOFF.OFS |
| Description | Payoff OFS |
| Source Type | Globus |
| Log File Dir | OFSLOG |
| Log details level | Full |
| Offline Queue | Y |
| Maint Msg Dets | Y |
| Det Prefix | PAYOFS |
| In Queue Dir | OFSIN |
| Syntax Type | OFS |
| Generic user | INPUTTER |
| Field Val | Yes |

### OVERRIDE

Create record in OVERRIDE with below detail

|  |  |
| --- | --- |
| Record id | Detail |
| AA-HUS.CASE.IS.MARKED | Case is marked with IN-Agreement |

### EB.ERROR

Create record in EB.ERROR with below detail

|  |  |
| --- | --- |
| Record id | Detail |
| AA-HUS.ID.MUST.BE.SYSTEM | ID must be system |
| AA-HUS.ALLOWED.INSTAL.FQC | Customer is not eligible for & Frequency |
| AA-HUS.CASE.IS.MARKED | Case is marked with IN-Agreement |

### AA.ACTIVITY

Create record in AA.ACTIVITY with below detail

|  |  |
| --- | --- |
| Field Name | Field Description |
| @ID | LENDING-CHARGE-FIREINSFEE |
| Description | Charge Fire Insurance Fee |
| Full Desc | Charge Fire Insurance Fee |
| Linked Activity | LENDING-CHANGE-FIREINSFEE |

### EB.HUS.GENERIC.PARAM

Create record in EB.HUS.GENERIC.PARAM with below details

|  |  |
| --- | --- |
| Field Name | Field Value |
| @ID | ALLOWED.INSTALL.FQC |
| Description | Allowed Frequency |
| PARAM.TYPE.1 | 1,3,4 |
| PARAM.NAME.1.1 | MONTH.FQC |
| PARAM.VALUE.1.1 | 1, 3, 6 |
| PARAM.TYPE.2 | 2,5,6,7,8 |
| PARAM.NAME.2.1 | MONTH.FQC |
| PARAM.VALUE.2.1 | 1, 3 |

# Technical Overview

## Architecture

NA

## Communications (Optional)

NA

# New Tables

All the below Table Structure has to be defined in EB.TABLE.DEFINITION table

## HB.INVOICE.PARAM

### Table Properties

|  |  |  |
| --- | --- | --- |
| No. | Property | Definition |
| 1 | Name | HB.INVOICE.PARAM |
| 2 | Product | EB |
| 3 | File Type | H |
| 4 | Classification | FIN |

### Table Layout

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **DESCRIPTION** | **MAX.CHAR** | **CHAR.TYPE** | **MV** | **Other Attributes/Validation** |
| @ID | @ID | 6 | Alphanumeric |  | A single record will exist with id SYSTEM |
| ACCOUNT.PROPERTY | ACCOUNT.PROPERTY | 30 | Alphanumeric | XX. |  |
| ACCOUNT.DESC | ACCOUNT.DESC | 30 | Alphanumeric |  |  |
| PRINCIPAL.INT.PROPERTY | PRINCIPAL.INT.PROPERTY | 30 | Alphanumeric | XX. |  |
| PRINCIPAL.INT.DESC | PRINCIPAL.INT.DESC | 30 | Alphanumeric |  |  |
| PENALTY.INT.PROPERTY | PENALTY.INT.PROPERTY | 30 | Alphanumeric | XX. |  |
| PENALTY.INT.DESC | PENALTY.INT.DESC | 30 | Alphanumeric |  |  |
| INVOICE.FEE.PROPERTY | INVOICE.FEE.PROPERTY | 30 | Alphanumeric | XX. |  |
| INVOICE.FEE.DESC | INVOICE.FEE.DESC | 30 | Alphanumeric |  |  |
| CHARGE.PROPERTY | CHARGE.PROPERTY | 30 | Alphanumeric | XX. |  |
| CHARGE.DESC | CHARGE.DESC | 30 | Alphanumeric |  |  |
| COST.PROPERTY | COST.PROPERTY | 30 | Alphanumeric | XX. |  |
| COST.DESC | COST.DESC | 30 | Alphanumeric |  |  |
| LEFTOVER.DESC | LEFTOVER.DESC | 30 | Alphanumeric |  |  |
| REMINDER.FEE.PROPERTY | REMINDER.FEE.PROPERTY | 30 | Alphanumeric | XX. |  |
| RPMT.ORDER.PRD.GROUP | RPMT.ORDER.PRD.GROUP | 30 | Alphanumeric | XX. |  |
| RPMT.ORDER.BALANCE | RPMT.ORDER.BALANCE | 30 | Alphanumeric | XX. |  |
| FEE.CHANNEL | FEE.CHANNEL | 30 | Alphanumeric | XX< |  |
| FEE.AMOUNT | FEE.AMOUNT | 13 | Amount | XX> |  |
| B2C.CUSTOMER.STATUS | B2C.CUSTOMER.STATUS | 5 | Alphanumeric | XX. |  |
| B2B.CUSTOMER.STATUS | B2B.CUSTOMER.STATUS | 5 | Alphanumeric | XX. |  |
| INVOICE.TYPE.CODE | INVOICE.TYPE.CODE | 5 | Alphanumeric |  |  |
| PAY.RQST.TYPE.CODE | PAY.RQST.TYPE.CODE | 5 | Alphanumeric |  |  |
| CR.NOTE.TYPE.CODE | CR.NOTE.TYPE.CODE | 5 | Alphanumeric |  |  |
| COMPANY.ID | COMPANY.ID | 30 | Alphanumeric |  |  |
| SCHEME.ID | SCHEME.ID | 30 | Alphanumeric |  |  |
| BANK.ACCOUNT | BANK.ACCOUNT | 30 | Alphanumeric |  |  |
| ADD.NAME | ADD.NAME | 30 | Alphanumeric |  |  |
| ADD.STREET | ADD.STREET | 30 | Alphanumeric |  |  |
| ADD.TOWN | ADD.TOWN | 30 | Alphanumeric |  |  |
| ADD.POST.CODE | ADD.POST.CODE | 30 | Alphanumeric |  |  |
| ADD.COUNTRY | ADD.COUNTRY | 30 | Alphanumeric |  |  |
| PAYMENT.MEANS.CODE | PAYMENT.MEANS.CODE | 5 | Alphanumeric |  |  |
| DIRECT.DEBIT.MEANS.CODE | DIRECT.DEBIT.MEANS.CODE | 5 | Alphanumeric |  |  |
| UNIT.CODE | UNIT.CODE | 5 | Alphanumeric |  |  |
| TAX.CATEGORY | TAX.CATEGORY | 5 | Alphanumeric |  |  |
| CUSTOMIZATION.ID | CUSTOMIZATION.ID | 65 | TEXT |  |  |
| PROFILE.ID | PROFILE.ID | 65 | TEXT |  |  |
| OVERDUE.TYPE | OVERDUE.TYPE | 30 | Alphanumeric | XX< |  |
| OVERDUE.DAYS | OVERDUE.DAYS | 5 | Alphanumeric | XX- |  |
| OVERDUE.NOTE | OVERDUE.NOTE | 65 | TEXT | XX> |  |
| PAYMENT.TOLERANCE.CCY | PAYMENT.TOLERANCE.CCY | 3 | CCY | XX< |  |
| PAYMENT.TOLERANCE | PAYMENT.TOLERANCE | 13 | Amount | XX> |  |
| INV.MIN.AMOUNT.CCY | INV.MIN.AMOUNT.CCY | 3 | CCY | XX< |  |
| INV.MIN.AMOUNT | INV.MIN.AMOUNT | 13 | Amount | XX> |  |
| NOTE.INVOICE | NOTE.INVOICE | 65 | TEXT |  |  |
| NOTE.PAYMENT.REQUEST | NOTE.PAYMENT.REQUEST | 65 | TEXT |  |  |
| NOTE.CREDIT.NOTE | NOTE.CREDIT.NOTE | 65 | TEXT |  |  |
| NOTE.COLLECTION | NOTE.COLLECTION | 65 | TEXT |  |  |
| RESERVED.10 | RESERVED.10 | 35 | Alphanumeric |  |  |
| RESERVED.9 | RESERVED.9 | 35 | Alphanumeric |  |  |
| RESERVED.8 | RESERVED.8 | 35 | Alphanumeric |  |  |
| RESERVED.7 | RESERVED.7 | 35 | Alphanumeric |  |  |
| RESERVED.6 | RESERVED.6 | 35 | Alphanumeric |  |  |
| RESEVED.5 | RESERVED.5 | 35 | Alphanumeric |  |  |
| RESERVED.4 | RESERVED.4 | 35 | Alphanumeric |  |  |
| RESERVED.3 | RESERVED.3 | 35 | Alphanumeric |  |  |
| RESERVED.2 | RESERVED.2 | 35 | Alphanumeric |  |  |
| RESERVED.1 | RESERVED.1 | 35 | Alphanumeric |  |  |
| LOCAL.REF | LOCAL.REF |  |  |  |  |

Note: EB.TABLE.DEFINITION>ADD.SPECIAL.FIELDS to be used to include LOCALREF

### Field Checks

NA

### Cross Validation Checks

NA

### Other Processing

**Check function:**

NA

1. **BEFORE.AUTH.WRITE:**

NA

1. **AFTER.AUTH.WRITE:**

NA

### INSERT Files

Automatically created based on the definition in EB.TABLE.DEFINITION

## HB.INVOICE.DETAILS

### Table Properties

|  |  |  |
| --- | --- | --- |
| No. | Property | Definition |
| 1 | Name | HB.INVOICE.DETAILS |
| 2 | Product | EB |
| 3 | File Type | H |
| 4 | Classification | FIN |

### Table Layout

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **DESCRIPTION** | **MAX.CHAR** | **CHAR.TYPE** | **MV** | **Other Attributes/Validation** |
| @ID | @ID | 50 | Alphanumeric |  |  |
| CASE.ID | CASE.ID | 30 | Alphanumeric |  |  |
| INITIAL.DUE.DATE | INITIAL.DUE.DATE | 8 | Date |  |  |
| CURRENCY | CURRENCY | 3 | CCY |  |  |
| CUSTOMER.ID | CUSTOMER.ID | 20 | Alphanumeric | XX< |  |
| REGISTER.ID | REGISTER.ID | 20 | Alphanumeric | XX- |  |
| DBT.REST.STATUS | DBT.REST.STATUS | 20 | Alphanumeric | XX> |  |
| CUSTOMER.TYPE | CUSTOMER.TYPE | 3 | Alphanumeric |  | **VETTING.TABLE:**  B2B B2C |
| CUSTOMER.REF | CUSTOMER.REF | 30 | Alphanumeric |  |  |
| ARRANGEMENT.ID | ARRANGEMENT.ID | 20 | ARR | XX< |  |
| LINE.ID | LINE.ID | 5 | Alphanumeric | XX-XX< |  |
| AMOUNT.TYPE | AMOUNT.TYPE | 30 | Alphanumeric | XX-XX- |  |
| AMOUNT | AMOUNT | 13 | Amount | XX-XX- |  |
| BILL.DUE.DATE | BILL.DUE.DATE | 8 | Date | XX-XX- |  |
| BILL.ID | BILL.ID | 20 | Alphanumeric | XX>XX> |  |
| TOTAL.AMOUNT | TOTAL.AMOUNT | 13 | Amount |  |  |
| ISSUE.DATE | ISSUE.DATE | 8 | Date |  |  |
| KID.NUMBER | KID.NUMBER | 30 | Alphanumeric |  |  |
| STATUS | STATUS | 30 | Alphanumeric |  | **VIRTUAL.TABLE:** HB.IN.DET.STATUS |
| STATUS.NARR | STATUS.NARR | 50 | Any |  |  |
| TYPE | TYPE | 30 | Alphanumeric |  | **VIRTUAL.TABLE:**  HB.IN.TYPE |
| DUE.DATE | DUE.DATE | 8 | Date |  |  |
| INVOICE.NO | INVOICE.NO | 30 | Alphanumeric |  |  |
| PREV.ISSUE.DATE | PREV.ISSUE.DATE | 8 | Date | XX< |  |
| PREV.KID.NUMBER | PREV.KID.NUMBER | 30 | Alphanumeric | XX- |  |
| PREV.STATUS | PREV.STATUS | 15 | Alphanumeric | XX- |  |
| PREV.STATUS.NARR | PREV.STATUS.NARR | 50 | Alphanumeric | XX- |  |
| PREV.TYPE | PREV.TYPE | 30 | Alphanumeric | XX- |  |
| PREV.DUE.DATE | PREV.DUE.DATE | 8 | Date | XX- |  |
| PREV.INVOICE.NO | PREV.INVOICE.NO | 30 | Alphanumeric | XX> |  |
| CHANNEL | CHANNEL | 30 | Alphanumeric |  | **VIRTUAL.TABLE:**  LA.INV.CHANNEL  Note: EB.LOOKUP record creation steps explained in TSD\_LA-Interim Workflows |
| PAYMENT.CHANNEL | PAYMENT.CHANNEL | 30 | Alphanumeric |  | **VIRTUAL.TABLE:**  HB.LA.PAY.CHL  Note: EB.LOOKUP record creation steps explained in TSD\_LA-Interim Workflows |
| DD.STATUS | DD.STATUS | 30 | Alphanumeric |  | **VETTING.TABLE:**  SENDING-CANCEL-DD  OK-CANCEL-DD  FAILED-CANCEL-DD  SENDING-CHANGE-DD-DUE-DATE  OK-CHANGE-DD-DUE-DATE  FAILED-CHANGE-DD-DUE-DATE |
| DD.STATUS.NARR | DD.STATUS.NARR | 30 | Alphanumeric |  |  |
| SEND.PMT.ADVICE | SEND.PMT.ADVICE | 1 | Alphanumeric |  | **VETTING.TABLE:**  Y |
| REASON.FOR.POSTPONEMENT | REASON.FOR.POSTPONEMENT | 65 | Alphanumeric |  |  |
| PREV.CO.REL.ID | Correlation id | 40 | Alphanumeric |  |  |
| POSTPONEMENT.DATE | PostponementDate |  | Date |  |  |
| PREMIUM.INFO | Premium Amt | 19 | Amount |  |  |
| DISCOUNT.INFO | Discount Amt | 19 | Amount |  |  |
| CUS.PREF.LANG | Preferred Language | 2 | Alphanumeric |  |  |
| RESERVED.10 | RESERVED.10 | 35 | Alphanumeric |  |  |
| RESERVED.9 | RESERVED.9 | 35 | Alphanumeric |  |  |
| RESERVED.8 | RESERVED.8 | 35 | Alphanumeric |  |  |
| RESERVED.7 | RESERVED.7 | 35 | Alphanumeric |  |  |
| RESERVED.6 | RESERVED.6 | 35 | Alphanumeric |  |  |
| RESEVED.5 | RESERVED.5 | 35 | Alphanumeric |  |  |
| RESERVED.4 | RESERVED.4 | 35 | Alphanumeric |  |  |
| RESERVED.3 | RESERVED.3 | 35 | Alphanumeric |  |  |
| RESERVED.2 | RESERVED.2 | 35 | Alphanumeric |  |  |
| RESERVED.1 | RESERVED.1 | 35 | Alphanumeric |  |  |
| LOCAL.REF | LOCAL.REF |  |  |  |  |

Note: EB.TABLE.DEFINITION>ADD.SPECIAL.FIELDS to be used to include LOCALREF

### Field Checks

NA

### Cross Validation Checks

NA

### Other Processing

**Check function:**

NA

1. **BEFORE.AUTH.WRITE:**

NA

1. **AFTER.AUTH.WRITE:**

NA

### INSERT Files

Automatically created based on the definition in EB.TABLE.DEFINITION

## HB.INVOICE.STAGING

### Table Properties

|  |  |  |
| --- | --- | --- |
| No. | Property | Definition |
| 1 | Name | HB.INVOICE.STAGING |
| 2 | Product | EB |
| 3 | File Type | H |
| 4 | Classification | FIN |

### Table Layout

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **DESCRIPTION** | **MAX.CHAR** | **CHAR.TYPE** | **MV** | **Other Attributes/Validation** |
| @ID | @ID | 30 | Alphanumeric |  |  |
| INVOICE.DETAILS.ID | INVOICE.DETAILS.ID | 30 | Alphanumeric | XX. |  |
| TYPE | TYPE | 30 | Alphanumeric |  | **VIRTUAL.TABLE:**  HB.IN.TYPE |
| RESEND | RESEND | 1 | Alphanumeric |  | **VETTING.TABLE:**  Y |
| CUSTOMER.ID | CUSTOMER.ID | 30 | Alphanumeric | XX< |  |
| REGISTER.ID | REGISTER.ID | 30 | Alphanumeric | XX- |  |
| DBT.REST.STATUS | DBT.REST.STATUS | 30 | Alphanumeric | XX> |  |
| CUSTOMER.TYPE | CUSTOMER.TYPE | 3 | Alphanumeric |  | **VETTING.TABLE:**  B2B B2C |
| CASE.ID | CASE.ID | 30 | Alphanumeric |  |  |
| FORCE.PAPER | FORCE.PAPER | 1 | Alphanumeric |  | **VETTING.TABLE:**  Y |
| CUSTOMIZATION.ID | CUSTOMIZATION.ID | 65 | TEXT |  |  |
| PROFILE.ID | PROFILE.ID | 65 | TEXT |  |  |
| FILE.ID | FILE.ID | 30 | Alphanumeric |  |  |
| ISSUE.DATE | ISSUE.DATE | 8 | Date |  |  |
| DUE.DATE | DUE.DATE | 8 | Date |  |  |
| INVOICE.TYPE.CODE | INVOICE.TYPE.CODE | 30 | Alphanumeric |  |  |
| CREDIT.NOTE.TYPE.CODE | CREDIT.NOTE.TYPE.CODE | 30 | Alphanumeric |  |  |
| DOCUMENT.CURRENCY.CODE | DOCUMENT.CURRENCY.CODE | 30 | Alphanumeric |  |  |
| BUYER.REFERENCE | BUYER.REFERENCE | 30 | Alphanumeric |  |  |
| INVOICE.DOCUMENT.REF | INVOICE.DOCUMENT.REF | 30 | Alphanumeric |  |  |
| CONTRACT.DOCUMENT.REF | CONTRACT.DOCUMENT.REF | 30 | Alphanumeric |  |  |
| S.END.POINT.ID | S.END.POINT.ID | 30 | Alphanumeric |  |  |
| S.SCHEME.ID | S.SCHEME.ID | 30 | Alphanumeric |  |  |
| S.NAME | S.NAME | 30 | Alphanumeric |  |  |
| S.STREET.NAME | S.STREET.NAME | 30 | Alphanumeric |  |  |
| S.CITY.NAME | S.CITY.NAME | 30 | Alphanumeric |  |  |
| S.POSTAL.ZONE | S.POSTAL.ZONE | 30 | Alphanumeric |  |  |
| S.COUNTRY.CODE | S.COUNTRY.CODE | 30 | Alphanumeric |  |  |
| S.REGISTRATION.NAME | S.REGISTRATION.NAME | 30 | Alphanumeric |  |  |
| S.COMPANY.ID | S.COMPANY.ID | 30 | Alphanumeric |  |  |
| S.COMPANY.SCHEME.ID | S.COMPANY.SCHEME.ID | 30 | Alphanumeric |  |  |
| B.END.POINT.ID | B.END.POINT.ID | 30 | Alphanumeric |  |  |
| B.SCHEME.ID | B.SCHEME.ID | 30 | Alphanumeric |  |  |
| B.NAME | B.NAME | 30 | Alphanumeric |  |  |
| B.STREET.NAME | B.STREET.NAME | 30 | Alphanumeric |  |  |
| B.ADD.STREET.NAME | B.ADD.STREET.NAME | 30 | Alphanumeric |  |  |
| B.CITY.NAME | B.CITY.NAME | 30 | Alphanumeric |  |  |
| B.POSTAL.ZONE | B.POSTAL.ZONE | 30 | Alphanumeric |  |  |
| B.COUNTRY.SUBENTITY | B.COUNTRY.SUBENTITY | 30 | Alphanumeric |  |  |
| B.COUNTRY.CODE | B.COUNTRY.CODE | 30 | Alphanumeric |  |  |
| B.REGISTRATION.NAME | B.REGISTRATION.NAME | 30 | Alphanumeric |  |  |
| PAYMENT.MEANS.CODE | PAYMENT.MEANS.CODE | 30 | Alphanumeric | XX< |  |
| PAYMENT.ID | PAYMENT.ID | 30 | Alphanumeric | XX- |  |
| PAYMENT.ACCOUNT | PAYMENT.ACCOUNT | 30 | Alphanumeric | XX> |  |
| CHARGE.INDICATOR | CHARGE.INDICATOR | 30 | Alphanumeric | XX< |  |
| CHARGE.REASON.CODE | CHARGE.REASON.CODE | 30 | Alphanumeric | XX- |  |
| CHARGE.REASON | CHARGE.REASON | 30 | Alphanumeric | XX- |  |
| CHARGE.AMOUNT | CHARGE.AMOUNT | 13 | Amount | XX- |  |
| CHARGE.CCY | CHARGE.CCY | 3 | CCY | XX- |  |
| CHR.BILL.DUE.DATE | BILL.DUE.DATE | 8 | Date | XX- |  |
| CHG.BILL.ID | BILL.ID | 20 | Alphanumeric | XX- |  |
| CHARGE.TAX.CATEGORY | CHARGE.TAX.CATEGORY | 30 | Alphanumeric | XX- |  |
| CHARGE.TAX.SCHEME.ID | CHARGE.TAX.SCHEME.ID | 30 | Alphanumeric | XX> |  |
| TAX.AMOUNT | TAX.AMOUNT | 15 | Amount | XX< |  |
| TAX.CCY | TAX.CCY | 3 | CCY | XX- |  |
| TAX.TAXABLE.AMT | TAX.TAXABLE.AMT | 13 | Amount | XX- |  |
| TAX.TAXABLE.CCY | TAX.TAXABLE.CCY | 3 | CCY | XX- |  |
| TAX.SUBTOT.AMT | TAX.SUBTOT.AMT | 13 | Amount | XX-XX< |  |
| TAX.SUBTOT.CCY | TAX.SUBTOT.CCY | 3 | CCY | XX-XX- |  |
| TAX.CATEGORY | TAX.CATEGORY | 30 | Alphanumeric | XX-XX- |  |
| TAX.EXEMPTION.CODE | TAX.EXEMPTION.CODE | 30 | Alphanumeric | XX-XX- |  |
| TAX.SCHEME.ID | TAX.SCHEME.ID | 30 | Alphanumeric | XX>XX> |  |
| T.LINE.EXTENSION.AMOUNT | T.LINE.EXTENSION.AMOUNT | 13 | Amount |  |  |
| T.LINE.EXTENSION.CCY | T.LINE.EXTENSION.CCY | 3 | CCY |  |  |
| T.TAX.EXCLUSIVE.AMOUNT | T.TAX.EXCLUSIVE.AMOUNT | 15 | Amount |  |  |
| T.TAX.EXCLUSIVE.CCY | T.TAX.EXCLUSIVE.CCY | 3 | CCY |  |  |
| T.TAX.INCLUSIVE.AMOUNT | T.TAX.INCLUSIVE.AMOUNT | 13 | Amount |  |  |
| T.TAX.INCLUSIVE.CCY | T.TAX.INCLUSIVE.CCY | 3 | CCY |  |  |
| T.CHARGE.TOT.AMOUNT | T.CHARGE.TOT.AMOUNT | 13 | Amount |  |  |
| T.CHARGE.TOT.CCY | T.CHARGE.TOT.CCY | 3 | CCY |  |  |
| T.PREPAID.AMOUNT | T.PREPAID.AMOUNT | 13 | Amount |  |  |
| T.PREPAID.CCY | T.PREPAID.CCY | 3 | CCY |  |  |
| T.PAYABLE.AMOUNT | T.PAYABLE.AMOUNT | 13 | Amount |  |  |
| T.PAYABLE.CCY | T.PAYABLE.CCY | 3 | CCY |  |  |
| L.ID | L.ID | 5 | Alphanumeric | XX< |  |
| L.NOTE | L.NOTE | 65 | TEXT | XX- |  |
| L.INVOICED.QUANTITY | L.INVOICED.QUANTITY | 30 | Alphanumeric | XX- |  |
| L.CREDITED.QUANTITY | L.CREDITED.QUANTITY | 30 | Alphanumeric | XX- |  |
| L.UNIT.CODE | L.UNIT.CODE | 30 | Alphanumeric | XX- |  |
| L.LINE.EXTENSION.AMOUNT | L.LINE.EXTENSION.AMOUNT | 13 | Amount | XX- |  |
| L.LINE.EXTENSION.CCY | L.LINE.EXTENSION.CCY | 3 | CCY | XX- |  |
| L.DOC.ID | L.DOC.ID | 30 | Alphanumeric | XX- |  |
| BILL.DUE.DATE | BILL.DUE.DATE | 8 | Date | XX- |  |
| BILL.ID | BILL.ID | 20 | Alphanumeric | XX- |  |
| L.DOC.SCHEME.ID | L.DOC.SCHEME.ID | 30 | Alphanumeric | XX- |  |
| L.DOC.TYPE.CODE | L.DOC.TYPE.CODE | 30 | Alphanumeric | XX- |  |
| L.NAME | L.NAME | 60 | Alphanumeric | XX- |  |
| L.TAX.CATEGORY.ID | L.TAX.CATEGORY.ID | 30 | Alphanumeric | XX- |  |
| L.TAX.SCHEME.ID | L.TAX.SCHEME.ID | 30 | Alphanumeric | XX- |  |
| L.PRICE.AMOUNT | L.PRICE.AMOUNT | 13 | Amount | XX- |  |
| L.PRICE.CCY | L.PRICE.CCY | 3 | CCY | XX> |  |
| NOTE | NOTE | 65 | Text | XX. |  |
| PREMIUM.INFO | PREMIUM.INFO | 19 | Amount |  |  |
| DISCOUNT.INFO | DISCOUNT.INFO | 19 | Amount |  |  |
| CUS.PREF.LANG | Preferred Language | 2 | Alphanumeric |  |  |
| RESERVED.10 | RESERVED.10 | 35 | Alphanumeric |  |  |
| RESERVED.9 | RESERVED.9 | 35 | Alphanumeric |  |  |
| RESERVED.8 | RESERVED.8 | 35 | Alphanumeric |  |  |
| RESERVED.7 | RESERVED.7 | 35 | Alphanumeric |  |  |
| RESERVED.6 | RESERVED.6 | 35 | Alphanumeric |  |  |
| RESEVED.5 | RESERVED.5 | 35 | Alphanumeric |  |  |
| RESERVED.4 | RESERVED.4 | 35 | Alphanumeric |  |  |
| RESERVED.3 | RESERVED.3 | 35 | Alphanumeric |  |  |
| RESERVED.2 | RESERVED.2 | 35 | Alphanumeric |  |  |
| RESERVED.1 | RESERVED.1 | 35 | Alphanumeric |  |  |
| LOCAL.REF | LOCAL.REF |  |  |  |  |

Note: EB.TABLE.DEFINITION>ADD.SPECIAL.FIELDS to be used to include LOCALREF

### Field Checks

NA

### Cross Validation Checks

NA

### Other Processing

**Check function:**

NA

1. **BEFORE.AUTH.WRITE:**

NA

1. **AFTER.AUTH.WRITE:**

NA

### INSERT Files

Automatically created based on the definition in EB.TABLE.DEFINITION

## HB.INVOICE.LEFTOVER

### Table Properties

|  |  |  |
| --- | --- | --- |
| No. | Property | Definition |
| 1 | Name | HB.INVOICE.LEFTOVER |
| 2 | Product | EB |
| 3 | File Type | H |
| 4 | Classification | FIN |

### Table Layout

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **DESCRIPTION** | **MAX.CHAR** | **CHAR.TYPE** | **MV** | **Other Attributes/Validation** |
| @ID | @ID | 30 | Alphanumeric |  |  |
| INVOICE.ID | INVOICE.ID | 30 | Alphanumeric | XX< |  |
| BILL.ID | BILL.ID | 30 | Alphanumeric | XX-XX |  |
| NEW.INVOICE | NEW.INVOICE | 30 | Alphanumeric | XX> |  |
| RESERVED.10 | RESERVED.10 | 35 | Alphanumeric |  |  |
| RESERVED.9 | RESERVED.9 | 35 | Alphanumeric |  |  |
| RESERVED.8 | RESERVED.8 | 35 | Alphanumeric |  |  |
| RESERVED.7 | RESERVED.7 | 35 | Alphanumeric |  |  |
| RESERVED.6 | RESERVED.6 | 35 | Alphanumeric |  |  |
| RESEVED.5 | RESERVED.5 | 35 | Alphanumeric |  |  |
| RESERVED.4 | RESERVED.4 | 35 | Alphanumeric |  |  |
| RESERVED.3 | RESERVED.3 | 35 | Alphanumeric |  |  |
| RESERVED.2 | RESERVED.2 | 35 | Alphanumeric |  |  |
| RESERVED.1 | RESERVED.1 | 35 | Alphanumeric |  |  |
| LOCAL.REF | LOCAL.REF |  |  |  |  |

Note: EB.TABLE.DEFINITION>ADD.SPECIAL.FIELDS to be used to include LOCALREF

### Field Checks

NA

### Cross Validation Checks

NA

### Other Processing

**Check function:**

NA

1. **BEFORE.AUTH.WRITE:**

NA

1. **AFTER.AUTH.WRITE:**

NA

### INSERT Files

Automatically created based on the definition in EB.TABLE.DEFINITION

## HB.CASE.INSTALMENTNO

### Table Properties

|  |  |  |
| --- | --- | --- |
| No. | Property | Definition |
| 1 | Name | HB.CASE.INSTALMENTNO |
| 2 | Product | EB |
| 3 | File Type | L |
| 4 | Classification | FIN |

### Table Layout

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **DESCRIPTION** | **MAX.CHAR** | **CHAR.TYPE** | **MV** | **Other Attributes/Validation** |
| @ID | @ID | 0030 | Alphanumeric |  |  |
| INSTAL.NO | INVOICE.ID | 003 | Alphanumeric |  |  |

## HB.INVOICE.UNPROCESS

### Table Properties

|  |  |  |
| --- | --- | --- |
| No. | Property | Definition |
| 1 | Name | HB.INVOICE.UNPROCESS |
| 2 | Product | EB |
| 3 | File Type | L |
| 4 | Classification | FIN |

### Table Layout

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **DESCRIPTION** | **MAX.CHAR** | **CHAR.TYPE** | **MV** | **Other Attributes/Validation** |
| @ID | @ID | 35 | Alphanumeric |  | Invoice Id |
| INVOICE.TYPE | INVOICE.TYPE | 3 | Alphanumeric |  |  |

### Field Checks

NA

### Cross Validation Checks

NA

### Other Processing

**Check function:**

NA

1. **BEFORE.AUTH.WRITE:**

NA

1. **AFTER.AUTH.WRITE:**

NA

### INSERT Files

Automatically created based on the definition in EB.TABLE.DEFINITION

## HB.UNPAID.INVOICE

### Table Properties

|  |  |  |
| --- | --- | --- |
| No. | Property | Definition |
| 1 | Name | HB.UNPAID.INVOICE |
| 2 | Product | EB |
| 3 | File Type | L |
| 4 | Classification | FIN |

### Table Layout

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **DESCRIPTION** | **MAX.CHAR** | **CHAR.TYPE** | **MV** | **Other Attributes/Validation** |
| @ID | @ID | 35 | Alphanumeric |  | Case id |
| INVOICE.DET.ID | INVOICE.DET.ID | 35 | Alphanumeric | XX. |  |

### Field Checks

NA

### Cross Validation Checks

NA

### Other Processing

**Check function:**

NA

1. **BEFORE.AUTH.WRITE:**

NA

1. **AFTER.AUTH.WRITE:**

NA

### INSERT Files

Automatically created based on the definition in EB.TABLE.DEFINITION

## HB.OVERDUE.REMINDER

### Table Properties

|  |  |  |
| --- | --- | --- |
| No. | Property | Definition |
| 1 | Name | HB.OVERDUE.REMINDER |
| 2 | Product | EB |
| 3 | File Type | L |
| 4 | Classification | FIN |

### Table Layout

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **DESCRIPTION** | **MAX.CHAR** | **CHAR.TYPE** | **MV** | **Other Attributes/Validation** |
| @ID | @ID | 35 | Alphanumeric |  | Case id |
| INV.DET.ID | INVOICE.DET.ID | 35 | Alphanumeric | XX. |  |
| FINAL.DEMAND | FINAL.DEMAND | 1 | Alphanumeric |  |  |

### Field Checks

NA

### Cross Validation Checks

NA

### Other Processing

**Check function:**

NA

1. **BEFORE.AUTH.WRITE:**

NA

1. **AFTER.AUTH.WRITE:**

NA

### INSERT Files

Automatically created based on the definition in EB.TABLE.DEFINITION

## HB.CASE.TODAY.BILLS

### Table Properties

|  |  |  |
| --- | --- | --- |
| No. | Property | Definition |
| 1 | Name | HB.CASE.TODAY.BILLS |
| 2 | Product | EB |
| 3 | File Type | L |
| 4 | Classification | FIN |

### Table Layout

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **DESCRIPTION** | **MAX.CHAR** | **CHAR.TYPE** | **MV** | **Other Attributes/Validation** |
| @ID | @ID | 35 | Alphanumeric |  | Case id |
| ArrangementId | ARRANGEMENT.ID | 35 | Alphanumeric | XX< |  |
| Activity Date | ACT.DATE | 8 | Date | XX> |  |

### Field Checks

NA

### Cross Validation Checks

NA

### Other Processing

**Check function:**

NA

1. **BEFORE.AUTH.WRITE:**

NA

1. **AFTER.AUTH.WRITE:**

NA

### INSERT Files

Automatically created based on the definition in EB.TABLE.DEFINITION

## HB.REPAYMENT.DETS

### Table Properties

|  |  |  |
| --- | --- | --- |
| No. | Property | Definition |
| 1 | Name | HB.REPAYMENT.DETS |
| 2 | Product | EB |
| 3 | File Type | L |
| 4 | Classification | FIN |

### Table Layout

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **DESCRIPTION** | **MAX.CHAR** | **CHAR.TYPE** | **MV** | **Other Attributes/Validation** |
| @ID | @ID | 35 | Alphanumeric |  | Case id |
| ArrangementId | ARRANGEMENT.ID | 35 | Alphanumeric | XX. |  |

### Field Checks

NA

### Cross Validation Checks

NA

### Other Processing

**Check function:**

NA

1. **BEFORE.AUTH.WRITE:**

NA

1. **AFTER.AUTH.WRITE:**

NA

### INSERT Files

Automatically created based on the definition in EB.TABLE.DEFINITION

## HB.ARR.TODAY.BILLS

### Table Properties

|  |  |  |
| --- | --- | --- |
| No. | Property | Definition |
| 1 | Name | HB.ARR.TODAY.BILLS |
| 2 | Product | EB |
| 3 | File Type | L |
| 4 | Classification | FIN |

### Table Layout

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **DESCRIPTION** | **MAX.CHAR** | **CHAR.TYPE** | **MV** | **Other Attributes/Validation** |
| @ID | @ID | 35 | Alphanumeric |  | Case id + Arrangement id |
| Activity Date | ACT.DATE | 8 | Date |  |  |

### Field Checks

NA

### Cross Validation Checks

NA

### Other Processing

**Check function:**

NA

1. **BEFORE.AUTH.WRITE:**

NA

1. **AFTER.AUTH.WRITE:**

NA

### INSERT Files

Automatically created based on the definition in EB.TABLE.DEFINITION

# Subroutines

## HusIdInvoiceParam

|  |  |
| --- | --- |
| Property | Specification |
| Type | S |
| Attached To | EB.TABLE.PROCEDURES>EB.HB.INVOICE.PARAM |
| Attached As | ID Routine |
| Dependency | NA |
| Description | ID routine to validate ID and raise an error message if it is other than SYSTEM |
| Arguments – IN | NA |
| Arguments – OUT | NA |
| Prelim Conditions |  |
| Subroutine Flow | 1. Get ID value from currentRecordId and if it is not equal to “SYSTEM” then raise an error message AA-HUS.ID.MUST.BE.SYSTEM |
| Special Instructions |  |

## HusInvoicePreparationLogic

|  |  |
| --- | --- |
| Property | Specification |
| Type | S |
| Attached To | AA.PRD.DES.ACTIVITY.API>MORTGAGE--20100101  Activity Class: LENDING-ISSUEBILL-PAYMENT.SCHEDULE  Property Class: PAYMENT.SCHEDULE  Action: ISSUE.BILL  Activity Class: LENDING-ISSUEBILL-CHARGE  Property Class: CHARGE  Action: ISSUE.BILL  Activity Class: LENDING-ISSUEBILL-PERIODIC.CHARGES  Property Class: PERIODIC.CHARGES  Action: ISSUE.BILL  Activity Class: LENDING-CAPTURE.BILL-BALANCE.MAINTENANCE  Property Class: PAYMENT.SCHEDULE  Action: UPDATE.SCHEDULES  Activity Class: LENDING-ADJUST.BILL-BALANCE.MAINTENANCE  Property Class: PAYMENT.SCHEDULE  Action: UPDATE.SCHEDULES  Activity Class: LENDING-ADJUST.INFO.BILL-BALANCE.MAINTENANCE  Property Class: BALANCE.MAINTENANCE  Action: DATA.CAPTURE |
| Attached As | Post Routine |
| Dependency | NA |
| Description | Post routine will get today’s bill details and update EB.HB.INVOICE.DETAILS |
| Arguments – IN | NA |
| Arguments – OUT | NA |
| Prelim Conditions |  |
| Subroutine Flow | 1. Get value from AA.ARR.ACCOUNT>HB.CASE.ID using AaPrdDesAccountRecord constructor and store it to the variable CaseIdVal 2. Get EFFECTIVE.DATE from arrangementActivityRecord and store it to the variable effDate 3. Read EB.HUS.LA.CASE record with @ID as HB.CASE.ID and get SEND.TO.COLLECTION field value 4. If SEND.TO.COLLECTION is equal to “YES” then exit the routine else continue further 5. Get current activity id from ArrangementContext by using getActivityId 6. If current activity id is equal to “LENDING-CHARGE-OVERDUE.FEE” or “LENDING-CHARGE-COLLECTION.FEE” then exit the routine else continue further 7. Get arrangement id from ArrangementContext by using getArrangementId and store it the variable ArrIdVal 8. Form EB.HB.ARR.TODAY.BILLS table id as below   arrTodayBillsId = CaseIdVal +"."+ ArrIdVal   1. Create new record in EB.HB.ARR.TODAY.BILLS with @ID as arrTodayBillsId and update ACT.DATE field with effDate |
| Special Instructions |  |

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

## HusAutUpdateInvId

|  |  |
| --- | --- |
| Property | Specification |
| Type | S |
| Attached To | VERSION.CONTROL>EB.HB.INVOICE.DETAILS |
| Attached As | Authorisation Routine |
| Dependency | NA |
| Description | Routine to update EB.HB.INVOICE.UNPROCESS and EB.HB.UNPAID.INVOICE table records based on STATUS & CHENNEL. |
| Arguments – IN | NA |
| Arguments – OUT | NA |
| Prelim Conditions |  |
| Subroutine Flow | 1. Extract the value before first dot(.) from currentRecordId and store it to the variable IdType 2. Get STATUS, CASE.ID and CHANNEL field value from currentRecord 3. If STATUS is equal to “NEW” and CHANNEL is not equal to empty then create record in EB.HB.INVOICE.UNPROCESS with @ID as currentRecordId and INVOICE.TYPE equal to IdType 4. If STATUS is not equal to “CLOSED” and IdType is equal to “I” then  * Read EB.HB.UNPAID.INVOICE record with @ID as CASE.ID * If record not found then create new record in EB.HB.UNPAID.INVOICE   @ID = CASE.ID  INVOICE.DET.ID = currentRecordId   * If record found in EB.HB.UNPAID.INVOICE then check currentRecordId is existing in EB.HB.UNPAID.INVOICE> INVOICE.DET.ID if it is not exist then add it in new multi-value  1. If STATUS is equal to “CLOSED” and IdType is equal to “I” then  * Read EB.HB.UNPAID.INVOICE record with @ID as CASE.ID * If record found in EB.HB.UNPAID.INVOICE then check currentRecordId is existing in EB.HB.UNPAID.INVOICE>INVOICE.DET.ID if it is exist then remove that multi-value. If it is last multi-value then delete the record. |
| Special Instructions |  |

## HusBatInvoiceGeneration

|  |  |
| --- | --- |
| Property | Specification |
| Type | S |
| Attached To | BATCH>BNK/HUS.B.INVOICE.GENERATION |
| Attached As | Batch Routine |
| Dependency | NA |
| Description | Routine will select records of EB.HB.INVOICE.DETAILS and will update EB.HB.INVOICE.STAGING to generate invoice. |
| Arguments – IN | NA |
| Arguments – OUT | NA |
| Prelim Conditions |  |
| Subroutine Flow | **Initialise**   1. Read EB.HB.INVOICE.PARAM table record with @ID as “SYSTEM” and get INV.MIN.AMOUNT field value 2. If INV.MIN.AMOUNT is null then consider it as Zero   **getIds:**   1. Select all records of EB.HB.INVOICE.UNPROCESS   **postUpdateRequest:**   1. Assign incoming id to the variable InvoiceDetId 2. Extract the value before first dot(.) from InvoiceDetId and store it to the variable IdType 3. If IdType equal to “I” then do the below steps    1. Read EB.HB.INVOICE.DETAILS details record with @ID as incoming id and get TOTAL.AMOUNT field value    2. If EB.HB.INVOICE.DETAILS>TOTAL.AMOUNT is greater than or equal to EB.HB.INVOICE.PARAM>INV.MIN.AMOUNT then do the steps in **UpdateStagingforIType** paragraph else do the steps in **UpdateLeftOver** paragraph 4. If IdType equal to “P” then create new record in EB.HB.INVOICE.STAGING with below details   Version = EB.HB.INVOICE.STAGING,HUS.OFS  Function = “INPUT”  Field mapping in the attached sheet     1. If IdType equal to “C” then create new record in EB.HB.INVOICE.STAGING with below details   Version = EB.HB.INVOICE.STAGING,HUS.OFS  Function = “INPUT”  Field mapping in the attached sheet     1. Delete record from EB.HB.INVOICE.UNPROCESS for the @ID stored in InvoiceDetId   **UpdateStagingforIType:**   1. Get EB.HB.INVOICE.DETAILS>CASE.ID and store it the variable CaseIdval 2. Read EB.HB.INVOICE.LEFTOVER record with @ID as CaseIdval 3. If record found in EB.HB.INVOICE.LEFTOVER, Loop through each multi-value set and do the below steps 4. If EB.HB.INVOICE.LEFTOVER>NEW.INVOICE is empty then get corresponding multi-value of BILL.ID 5. Read AA.BILL.DETAILS record with @ID as BILL.ID and get OS.TOTAL.AMOUNT and add new multi-value in EB.HB.INVOICE.DETAILS with the below mapping   ARRANGEMENT.ID = “”(Blank)  LINE.ID = Get value from previous multi-value set and increment by 1  AMOUNT.TYPE = EB.HB.INVOICE.PARAM> LEFTOVER.DESC  AMOUNT = AA.BILL.DETAILS>OS.TOTAL.AMOUNT  BILL.ID = “” (Blank)  TOTAL.AMOUNT = Sum of all amount fields.   1. Update current multi-value of EB.HB.INVOICE.LEFTOVER>NEW.INVOICE with the value of InvoiceDetId 2. Create new record in EB.HB.INVOICE.STAGING with below details   Version = EB.HB.INVOICE.STAGING,HUS.OFS  Function = “INPUT”  Field mapping in the attached sheet    **UpdateLeftOver:**   1. Get EB.HB.INVOICE.DETAILS>CASE.ID and store it the variable CaseIdval 2. Read EB.HB.INVOICE.LEFTOVER record with @ID as CaseIdval 3. If record not found then create new record in EB.HB.INVOICE.LEFTOVER with below field mapping else add new multi-value set.   INVOICE.ID = KID.NUMBER  BILL.ID = EB.HB.INVOICE.DETAILS>BILL.ID (All Multi-value)   1. Update EB.HB.INVOICE.DETAILS>STATUS with “CLOSED”   Version = EB.HB.INVOICE.DETAILS,HUS.OFS  Function = “INPUT” |
| Special Instructions |  |

## HusBatOverdueProcess

|  |  |
| --- | --- |
| Property | Specification |
| Type | S |
| Attached To | BATCH>BNK/HUS.B.OVERDUE.PROCESS |
| Attached As | Batch Routine |
| Dependency | NA |
| Description | This routine used to monitor and update the overdue status of the invoices |
| Arguments – IN | NA |
| Arguments – OUT | NA |
| Prelim Conditions |  |
| Subroutine Flow | **Initialise:**   1. Read EB.HB.INVOICE.PARAM table record with @ID as “SYSTEM” and get OVERDUE.DAYS and OVERDUE.TYPE multi-value   **getIds:**   1. Select all records of EB.HB.UNPAID.INVOICE   **postUpdateRequest:**   1. Assign incoming id to the variable CaseIdval 2. Read EB.HUS.LA.CASE with @ID as CaseIdval and get PROJECT.PARTY multi-value 3. Calculate count of PROJECT.PARTY multi-value and store it to the variable CaseCustCnt 4. Initialise NegotiationCnt = 0 and AdoptedCnt = 0 5. Loop … through each PROJECT.PARTY multi-value and do the below steps  * Read CUSTOMER record with @ID as PROJECT.PARTY and get DBT.REST.STATUS field value * If CUSTOMER>DBT.REST.STATUS is equal to “NEGOTIATION” then increment NegotiationCnt by 1 * If CUSTOMER>DBT.REST.STATUS is equal to “ADOPTED” then increment AdoptedCnt by 1  1. If CaseCustCnt is not equal to NegotiationCnt then continue further 2. Read EB.HB.INVOICE.PARAM record with @ID as “SYSTEM” and get OVERDUE.DAYS and OVERDUE.TYPE multi-value 3. Read EB.HB.UNPAID.INVOICE record with @ID as CaseIdval and get INVOICE.DET.ID multi-value 4. Loop … through each INVOICE.DET.ID multi-value and do the below steps  * Read EB.HB.INVOICE.DETAILS record with @ID as INVOICE.DET.ID * Get EB.HB.INVOICE.DETAILS>DUE.DATE field value * Count the number of days between DUE.DATE and TODAY and store the result to OverDueDays * Initialise OverdueType = “” * Loop … through each EB.HB.INVOICE.PARAM>OVERDUE.DAYS and OVERDUE.TYPE and do the below steps * Using substring remove days from OVERDUE.DAYS and store the result to the variable ParamDays   (Example:  OVERDUE.DAYS =14D  ParamDays = 14)   * If OverDueDays is greater than ParamDays then assign the corresponding multi-value of OVERDUE.TYPE to OverdueType * If OverdueType is not equal to empty, do the steps in **OverdueUpdate** paragraph   **OverdueUpdate:**   1. If OverdueType is equal to “FINAL.DEMAND.PAYMENT” and CaseCustCnt is equal to AdoptedCnt then don’t continue further 2. If OverdueType is equal to EB.HB.INVOICE.DETAILS>TYPE then don’t continue further 3. Read EB.HB.OVERDUE.REMINDER record with @ID as CaseIdval  * If record not found then create new record in EB.HB.OVERDUE.REMINDER   @ID = CASE.ID  INV.DET.ID = EB.HB.UNPAID.INVOICE>INVOICE.DET.ID  FINAL.DEMAND = If OverdueType is equal to “FINAL.DEMAND.PAYMENT” then “Y”   * If record found in EB.HB.OVERDUE.REMINDER then check EB.HB.UNPAID.INVOICE>INVOICE.DET.ID is existing in EB.HB.OVERDUE.REMINDER>INV.DET.ID if it is not exist then add it in new multi-value and update FINAL.DEMAND field with “Y” If OverdueType is equal to “FINAL.DEMAND.PAYMENT”  1. Add 14 days to TODAY and store the resultant value to NewDueDate 2. If NewDueDate is holiday then cycle it to next working day. 3. Assign TODAY date to HighMatDate variable and initialize InvoiceArrId = “” 4. Get EB.HB.INVOICE.DETAILS>ARRANGEMENT.ID multi-value and Loop through each value and do the below steps  * Read AA.ACCOUNT.DETAILS record and get MATURITY.DATE field value * If MATURITY.DATE is greater than HighMatDate then assign MATURITY.DATE to HighMatDate variable and ARRANGEMENT.ID to InvoiceArrId  1. Update EB.HB.INVOICE.DETAILS record with below mapping   Version = EB.HB.INVOICE.DETAILS,HUS.OFS  Function = “INPUT”  @ID = INVOICE.DET.ID  Open new multi-value  PREV.ISSUE.DATE = EB.HB.INVOICE.DETAILS> ISSUE.DATE (Old record value)  PREV.KID.NUMBER = EB.HB.INVOICE.DETAILS> KID.NUMBER (Old record value)  PREV.STATUS = EB.HB.INVOICE.DETAILS> STATUS(Old record value)  PREV.STATUS.NARR = EB.HB.INVOICE.DETAILS>STATUS.NARR(Old record value)  PREV.TYPE = EB.HB.INVOICE.DETAILS>TYPE(Old record value)  PREV.DUE.DATE = EB.HB.INVOICE.DETAILS>DUE.DATE(Old record value)  PREV.INVOICE.NO = EB.HB.INVOICE.DETAILS>INVOICE.NO(Old record value)  Update below field with below mapping  STATUS = NEW  ISSUE.DATE = TODAY  DUE.DATE = NewDueDate  TYPE = OverdueType  Locate InvoiceArrId in EB.HB.INVOICE.DETAILS> ARRANGEMENT.ID and expand one sub-value set under this multi-value  LINE.ID = Get value from previous sub-value of LINE.ID and increment by 1 (Also change subsequent multi/Sub value set)  AMOUNT.TYPE = “REMINDER.FEE”  If OverdueType is equal to “DEBT.COLLECTION.NOTICE” then update AMOUNT = AA.PRD.DES.CHARGE>FIXED.AMOUNT from COLLECTIONFEE property condition  If OverdueType is equal to “FINAL.DEMAND.PAYMENT” then update AMOUNT = AA.PRD.DES.CHARGE>FIXED.AMOUNT from OVERDUEFEE property condition  BILL.ID = “” (Empty)   1. If OverdueType is equal to “DEBT.COLLECTION.NOTICE” then trigger LENDING-CHARGE-COLLECTION.FEE activity for the arrangement id in InvoiceArrId variable 2. If OverdueType is equal to “FINAL.DEMAND.PAYME” then trigger LENDING-CHARGE-OVERDUE.FEE activity for the arrangement id in InvoiceArrId variable |
| Special Instructions |  |

## HusAaPostUpdInvoice

|  |  |
| --- | --- |
| Property | Specification |
| Type | S |
| Attached To | AA.PRD.DES.ACTIVITY.API>MORTGAGE--20100101  Activity Class: LENDING-APPLYPAYMENT-PAYMENT.RULES  Property Class: PAYMENT.SCHEDULE  Action: CANCEL.BILLS |
| Attached As | Post Routine |
| Dependency | NA |
| Description | Routine to update EB.HB.INVOICE.DETAILS & EB.HB.INVOICE.LEFTOVER table records based on payment tolerance |
| Arguments – IN | NA |
| Arguments – OUT | NA |
| Prelim Conditions |  |
| Subroutine Flow | 1. Get account id using getLinkedAccount from arrangementContext and store it to the variable AccountId 2. Read ACCOUNT record with @ID as AccountId and get HB.CASE.ID and store it to the variable CaseId 3. Get arrangement id from arrangementContext and store it the variable arrIdVal 4. Read EB.HB.CASE.TODAY.BILLS record with @ID as CaseIdVal 5. If record exist then add new multi-value and update ARRANGEMENT.ID field with arrIdVal 6. If record does not exist then create new record with @ID as CaseIdVal and update ARRANGEMENT.ID field with ArrIdVal |
| Special Instructions |  |

## HusAaPostCancelDd

|  |  |
| --- | --- |
| Property | Specification |
| Type | S |
| Attached To | AA.PRD.DES.ACTIVITY.API>MORTGAGE--20100101  Activity: LENDING-APPLYPAYMENT-PR.PAYOFF  Property Class: INTEREST  Action: LINK.UPDATE |
| Attached As | Post Routine |
| Dependency | NA |
| Description | Routine to update EB.HB.INVOICE.DETAILS>DD.STATUS with “SENDING-CANCEL-DD” when the loan is paid off |
| Arguments – IN | NA |
| Arguments – OUT | NA |
| Prelim Conditions |  |
| Subroutine Flow | 1. Get account id using getLinkedAccount from arrangementContext and store it to the variable AccountId 2. Read ACCOUNT record with @ID as AccountId and get HB.CASE.ID and store it to the variable CaseId 3. Read EB.HUS.LA.CASE record with @ID as CaseId and get all ARRANGEMENT.ID multi-value 4. Initialize ActiveArrCnt = 0 5. Loop … through each ARRANGEMENT.ID and do the below steps  * Read AA.ARRANGEMENT record and get ARR.STATUS field value * If ARR.STATUS is not equal “CLOSE” or “MATURED” or “PENDING.CLOSURE” then increment ActiveArrCnt by 1  1. If ActiveArrCnt greater than 1 then exit the routine else continue further 2. Read EB.HB.UNPAID.INVOICE record with @ID as CaseId and loop … through each INVOICE.DET.ID and do the below steps  * Read EB.HB.INVOICE.DETAILS record with @ID as INVOICE.DET.ID * If EB.HB.INVOICE.DETAILS >PAYMENT.CHANNEL is equal to “DIRECT.DEBIT” then update EB.HB.INVOICE.DETAILS record with below mapping   Version = EB.HB.INVOICE.DETAILS,HUS.DD.CANCEL  Function = “INPUT”  DD.STATUS = “SENDING-CANCEL-DD” |
| Special Instructions |  |

## HusBatReminderInvoice

|  |  |
| --- | --- |
| Property | Specification |
| Type | S |
| Attached To | BATCH>BNK/HUS.B.REMINDER.PROCESS |
| Attached As | Batch Routine |
| Dependency | NA |
| Description | This routine used to send reminder invoice |
| Arguments – IN | NA |
| Arguments – OUT | NA |
| Prelim Conditions |  |
| Subroutine Flow | **getIds:**   1. Select all records of EB.HB.OVERDUE.REMINDER   **postUpdateRequest:**   1. Assign incoming id to the variable CaseIdval 2. Read EB.HB.OVERDUE.REMINDER record with @ID as CaseIdval 3. If EB.HB.OVERDUE.REMINDER>FINAL.DEMAND is equal to “Y” then do the steps in **FinalDemandPrcess** paragraph else do the steps in **DebtCollectionProcess** paragraph   **DebtCollectionProcess:**   1. Get EB.HB.OVERDUE.REMINDER>INV.DET.ID multi-value 2. Loop … through each INV.DET.ID and do the below steps  * Read EB.HB.INVOICE.DETAILS record with @ID as INV.DET.ID * Create new record in EB.HB.INVOICE.STAGING with below details   Version = EB.HB.INVOICE.STAGING,HUS.OFS  Function = “INPUT”  Field mapping in the attached sheet     * Remove respective INV.DET.ID multi-value from EB.HB.OVERDUE.REMINDER. If it is last multi-value then delete the record.   **FinalDemandPrcess:**   1. Get EB.HB.OVERDUE.REMINDER>INV.DET.ID multi-value 2. Loop … through each INV.DET.ID and do the below steps  * Read EB.HB.INVOICE.DETAILS record with @ID as INV.DET.ID * Create/update record in EB.HB.INVOICE.STAGING with below details   Version = EB.HB.INVOICE.STAGING,HUS.OFS  Function = “INPUT”  Field mapping in the attached sheet    Only multi-value to be updated from the second record.   1. Delete record from EB.HB.OVERDUE.REMINDER for the @ID in CaseIdval |
| Special Instructions |  |

## HusBUpdCaseToday

|  |  |
| --- | --- |
| Property | Specification |
| Type | S |
| Attached To | BATCH> BNK/HusBUpdCaseToday (Stage: A101)  BATCH> BNK/HusBUpdCaseTodaySod (Stage: D701) |
| Attached As | Batch Routine |
| Dependency | NA |
| Description | This routine used to update EB.HB.CASE.TODAY.BILLS |
| Arguments – IN | NA |
| Arguments – OUT | NA |
| Prelim Conditions |  |
| Subroutine Flow | **processSingleThreaded:**   1. Select all records of EB.HB.ARR.TODAY.BILLS 2. Loop through each id from EB.HB.ARR.TODAY.BILLS and do the below steps  * Get case id and arrangement id from @id of EB.HB.ARR.TODAY.BILLS as shown below   caseId = id.split("\\.")[0];  arrnId = id.split("\\.")[1];   * Read EB.HB.ARR.TODAY.BILLS and get ACT.DATE field value and store it the variable effDate * Read EB.HB.CASE.TODAY.BILLS record with @ID as caseId * If record exist then add new multi-value and update ARRANGEMENT.ID field with arrnId and ACT.DATE field with value effDate * If record does not exist then create new record with @ID as CaseIdVal and update ARRANGEMENT.ID field with ArrIdVal and ACT.DATE field with value effDate |
| Special Instructions |  |

## HusAncStatusUpd

|  |  |
| --- | --- |
| Property | Specification |
| Type | S |
| Attached To | VERSION>EB.HB.INVOICE.DETAILS,HUS.RECALL |
| Attached As | Auto new content routine |
| Dependency | NA |
| Description | Routine to update STAUS and DD.STATUS field |
| Arguments – IN | NA |
| Arguments – OUT | NA |
| Prelim Conditions |  |
| Subroutine Flow | 1. If DD.STATUS is equal to “FAILED-CANCEL-DD” then update DD.STATUS field with value “SENDING-CANCEL-DD” 2. If DD.STATUS is equal to “FAILED-CHANGE-DD-DUE-DATE” then update DD.STATUS field with value “SENDING-CHANGE-DD-DUE-DATE” 3. If STATUS is equal to “FAILED-GET-CHANNEL” then update STATUS field with value “SENDING-GET-CHANNEL” 4. If STATUS is equal to “FAILED” then update STATUS field with value “SENDING” |
| Special Instructions |  |

## HusAuthSimCapConcUpd

|  |  |
| --- | --- |
| Property | Specification |
| Type | S |
| Attached To | VERSION>AA.SIMULATION.CAPTURE,AA.DRILL.PAYOFF |
| Attached As | Authorisation Routine |
| Dependency | NA |
| Description | This Authorization routine will update Arrangement id under EB.HB.CASE.TODAY.BILLS table for HB.CASE.ID |
| Arguments – IN | NA |
| Arguments – OUT | NA |
| Prelim Conditions |  |
| Subroutine Flow | 1. Get EFFECTIVE.DATE and ARRANGEMENT from currentRecord and store it to the variable effData and ArrIdVal 2. Get value from AA.SIM.ACCOUNT>HB.CASE.ID using AaPrdDesAccountRecord constructor and store it to the variable CaseIdVal 3. Read EB.HB.CASE.TODAY.BILLS record with @ID as CaseIdVal 4. If record exist then add new multi-value and update ARRANGEMENT.ID field with ArrIdVal and ACT.DATE field with value effData 5. If record does not exist then create new record with @ID as CaseIdVal and update ARRANGEMENT.ID field with ArrIdVal and ACT.DATE field with value effData |
| Special Instructions |  |

## HusBatAaUpdInvoice

|  |  |
| --- | --- |
| Property | Specification |
| Type | S |
| Attached To | BATCH>BNK/HUS.B.REPAYMENT.UPD.INVOICE |
| Attached As | Batch Routine |
| Dependency | NA |
| Description | This routine used to update EB.HB.INVOICE.DETAILS & EB.HB.INVOICE.LEFTOVER table records based on payment tolerance |
| Arguments – IN | NA |
| Arguments – OUT | NA |
| Prelim Conditions |  |
| Subroutine Flow | **getIds:**   1. Select all records of EB.HB.REPAYMENT.DETS   **postUpdateRequest:**   1. Assign incoming id to the variable CaseId 2. Read EB.HB.INVOICE.PARAM record with @ID as “SYSTEM” and get PAYMENT.TOLERANCE.CCY and PAYMENT.TOLERANCE field value 3. Read EB.HUS.LA.CASE record with @ID as CaseId and get arrangement id from first multi-value of ARRANGEMENT.ID and store it to the variable arrIdval 4. Read AA.ARRANGEMENT record with @ID as arrIdval and get CURRENCY field value and store it the variable ArrCcyVal 5. Locate ArrCcyVal in EB.HB.INVOICE.PARAM>PAYMENT.TOLERANCE.CCY and get the corresponding PAYMENT.TOLERANCE field value 6. If PAYMENT.TOLERANCE is empty then consider it as Zero 7. Read EB.HB.UNPAID.INVOICE record with @ID as CaseId and loop … through each INVOICE.DET.ID multi-value and do the below steps  * Initialize TotalOutstandingAmt = 0 * Initialize BillIdList = “” * Read EB.HB.INVOICE.DETAILS record with @Id as INVOICE.DET.ID * Get all BILL.ID from EB.HB.INVOICE.DETAILS * Loop … through each BILL.ID and Read AA.BILL.DETAILS for each bills and get OS.TOTAL.AMOUNT * If OS.TOTAL.AMOUNT is greater than zero then add OS.TOTAL.AMOUNT to TotalOutstandingAmt variable and append BILL.ID to List variable BillIdList * If TotalOutstandingAmt is greater than zero and lesser than PAYMENT.TOLERANCE then * Read EB.HB.INVOICE.LEFTOVER record with @ID as CaseId * If record not found then create new record in EB.HB.INVOICE.LEFTOVER with below field mapping else add new multi-value set in the existing record.   Loop … through each value from InvoiceBillList and add new multi-value set  INVOICE.ID = INVOICE.DET.ID (Current EB.HB.UNPAID.INVOICE>INVOICE.DET.ID extracted in a loop)  BILL.ID = Extract each Bill id from BillIdList variable and insert it in new sub-value.  Update EB.HB.INVOICE.DETAILS>STATUS with “CLOSED”  Version = EB.HB.INVOICE.DETAILS,HUS.OFS  Function = “INPUT”  @ID = INVOICE.DET.ID (Current EB.HB.UNPAID.INVOICE>INVOICE.DET.ID extracted in a loop)  STATUS = “CLOSED” |
| Special Instructions |  |

## HusDefChannelInvUpd

|  |  |
| --- | --- |
| Property | Specification |
| Type | S |
| Attached To | VERSION>EB.HB.INVOICE.DETAILS,HUS.OFS, EB.HB.INVOICE.DETAILS,HUS.LS.OFS and EB.HB.INVOICE.DETAILS,HUS.INPUT |
| Attached As | Default Routine |
| Dependency | NA |
| Description | This Default routine used to default PAYMENT.CHANNEL & CHANNEL field values in EB.HB.INVOICE.DETAILS record from EB.HUS.LA.CASE |
| Arguments – IN | NA |
| Arguments – OUT | NA |
| Prelim Conditions |  |
| Subroutine Flow | 1. Get CASE.ID from currentRecord and store it to the variable caseIdVal 2. Read EB.HUS.LA.CASE record with @ID as caseIdVal and get PAYMENT.CHANNEL and INVOICE.CHANNEL field and store it the variable payChannel and InvChannel respectively 3. If InvChannel is not empty then update CHANNEL field of current record with value InvChannel else update CHANNEL field of current record with value “PAPER” 4. update PAYMENT.CHANNEL field of current record with value payChannel |
| Special Instructions |  |

## HusBulIntRecall

|  |  |
| --- | --- |
| Property | Specification |
| Type | S |
| Attached To | ENQUIRY>HUS.INTERFACE.RECALL |
| Attached As | Build routine |
| Dependency | NA |
| Description | Routine to add selection values to STAUS and DD.STATUS |
| Arguments – IN | NA |
| Arguments – OUT | NA |
| Prelim Conditions |  |
| Subroutine Flow | 1. Clear filterCriteria argument using filterCriteria.clear 2. Using setFieldname add “STATUS” to fieldname in filterCriteria 3. Using setOperand add “EQ” to Operand in filterCriteria 4. Using setValue add “FAILED-GET-CHANNEL FAILED” to Value in filterCriteria 5. Using setFieldname add “DD.STATUS” to fieldname in filterCriteria 6. Using setOperand add “EQ” to Operand in filterCriteria 7. Using setValue add “FAILED-CANCEL-DD FAILED-CHANGE-DD-DUE-DATE” to Value in filterCriteria |
| Special Instructions |  |

## HusRepaymentApi

|  |  |
| --- | --- |
| Property | Specification |
| Type | S |
| Attached To | NA |
| Attached As | Call routine |
| Dependency | NA |
| Description | Routine to return arrangement ids and amounts to allocate to each arrangement |
| Arguments – IN | KIDNumber, CaseNumber and TotalAmount |
| Arguments – OUT | ArrangementIds, AllocAmount, PayTypes, UnAllocatedAmt |
| Prelim Conditions |  |
| Subroutine Flow | 1. Get Kidnumber and CaseNumber from argument if both are null then return error message “Kid Number or Case Number is mandatory” and exit the routine 2. Get TotalAmount from argument, If it is null or less than or equal to zero then return error message “Must be valid Amount” and exit the routine 3. Read EB.HB.INVOICE.PARAM record with @ID “SYSTEM” and get field value of RPMT.ORDER.PRD.GROUP, RPMT.ORDER.BALANCE, ACCOUNT.PROPERTY, PRINCIPAL.INT.PROPERTY, PENALTY.INT.PROPERTY, CHARGE.PROPERTY and COST.PROPERTY 4. Create List array named ProduGrpList and store EB.HB.INVOICE.PARAM>RPMT.ORDER.PRD.GROUP as it was in the same order. 5. If CaseNumber is equal to null then  * Read EB.HB.INVOICE.STAGING table record with @ID as Kidnumber and get first multi-value of INVOICE.DETAILS.ID * Read EB.HB.INVOICE.DETAILS record with @ID as INVOICE.DETAILS.ID and get CASE.ID and store it to the variable CaseNoVal * If CaseNumber from incoming argument is not null then store it to the variable CaseNoVal  1. Read EB.HUS.LA.CASE record with @ID as CaseNoVal 2. If record not found in EB.HUS.LA.CASE then return error message “Invalid Case Id” and exit the routine 3. Using Loop statement, extract each multi-value from EB.HUS.LA.CASE>ARRANGEMENT.ID and store it to the variable ArrangementIdVal and do the below steps  * Read AA.ARRANGEMENT record with @ID as ArrangementIdVal and get PRODUCT.GROUP and store it to the variable ProdGrp * Call getBillIdsForSettlementStatus to get UNPAID bill Ids * Loop though each Bill id’s and do all the steps explained in **ArrayFormation** paragraph  1. Using split & Loop statement remove each value from UnpaidArrList and create multiple list array based on the product group names in EB.HB.INVOICE.PARAM>RPMT.ORDER.PRD.GROUP   **Example:**  Before Loop statement:  HUS.MORTGAGES\*20200101\*ACCOUNT\*AA20001\*2000\*R  HUS.MORTGAGES\*20200101\*PRINCIPALINT\*AA20001\*50\*R  HUS.MORTGAGES\*20200301\*CHARGE\*AA20001\*20\*R  HUS.GRANTS\*20200101\*ACCOUNT\*AA20004\*2000\*P  HUS.MORTGAGES\*20200201\*PENALTYINT\*AA20002\*40\*R  HUS.MORTGAGES\*20200301\*ACCOUNT\*AA20002\*1500\*R  HUS.GRANTS\*20200201\*ACCOUNT\*AA20007\*1000\*P  HUS.MORTGAGES\*20200101\*PRINCIPALINT\*AA20003\*25\*R  HUS.MORTGAGES\*20200101\*COST\*AA20003\*20\*R  After Loop statement:  First List Array:  HUS.MORTGAGES\*20200101\*ACCOUNT\*AA20001\*2000\*R  HUS.MORTGAGES\*20200101\*PRINCIPALINT\*AA20001\*50\*R  HUS.MORTGAGES\*20200301\*CHARGE\*AA20001\*20\*R  HUS.MORTGAGES\*20200201\*PENALTYINT\*AA20002\*40\*R  HUS.MORTGAGES\*20200301\*ACCOUNT\*AA20002\*1500\*R  HUS.MORTGAGES\*20200101\*PRINCIPALINT\*AA20003\*25\*R  HUS.MORTGAGES\*20200101\*COST\*AA20003\*20\*R  Second List Array:  HUS.GRANTS\*20200101\*ACCOUNT\*AA20004\*2000\*P  HUS.GRANTS\*20200201\*ACCOUNT\*AA20007\*1000\*P   1. Using collection.sort, split and loop statement, process each List array and sort the array based on the payment date & Property in order of EB.HB.INVOICE.PARAM>RPMT.ORDER.BALANCE   **Example:**  Before Sorting:  First List Array:  HUS.MORTGAGES\*20200101\*ACCOUNT\*AA20001\*2000\*R  HUS.MORTGAGES\*20200101\*PRINCIPALINT\*AA20001\*50\*R  HUS.MORTGAGES\*20200301\*CHARGE\*AA20001\*20\*R  HUS.MORTGAGES\*20200201\*PENALTYINT\*AA20002\*40\*R  HUS.MORTGAGES\*20200301\*ACCOUNT\*AA20002\*1500\*R  HUS.MORTGAGES\*20200101\*PRINCIPALINT\*AA20003\*25\*R  HUS.MORTGAGES\*20200101\*COST\*AA20003\*20\*R  Second List Array:  HUS.GRANTS\*20200101\*ACCOUNT\*AA20004\*2000\*P  HUS.GRANTS\*20200201\*ACCOUNT\*AA20007\*1000\*P  After Sorting:  First List Array:  HUS.MORTGAGES\*20200101\*COST\*AA20003\*20\*R  HUS.MORTGAGES\*20200101\*PRINCIPALINT\*AA20001\*50\*R  HUS.MORTGAGES\*20200101\*PRINCIPALINT\*AA20003\*25\*R  HUS.MORTGAGES\*20200101\*ACCOUNT\*AA20001\*2000\*R  HUS.MORTGAGES\*20200201\*PENALTYINT\*AA20002\*40\*R  HUS.MORTGAGES\*20200301\*CHARGE\*AA20001\*20\*R  HUS.MORTGAGES\*20200301\*ACCOUNT\*AA20002\*1500\*R  Second List Array:  HUS.GRANTS\*20200101\*ACCOUNT\*AA20004\*2000\*P  HUS.GRANTS\*20200201\*ACCOUNT\*AA20007\*1000\*P   1. Using Loop statement, remove each line from List arrays (First array, second array and so on) and do the below steps  * Using Split, extract Arrrangement id, Amount and paytype * If extracted amount is less than TotalAmount (Incoming argument) * Reduce the extracted amount from TotalAmount   Example: TotalAmount = 1000  Extracted Amount = 200  Calculated new TotalAmount = 800   * Do the steps in **FinalArrayFormation** paragraph * If extracted amount is greater than TotalAmount then * Assign extracted amount with TotalAmount * Do the steps in **FinalArrayFormation** paragraph * Assign TotalAmount = 0  1. If TotalAmount is greater than zero then assign TotalAmount value to UnAllocatedAmt in the first position (Outgoing Arugument). Get highest maturity date arrangement from EB.HUS.LA.CASE and assign it to UnAllocatedAmt in the second position   **ArrayFormation:**   1. Read AA.BILL.DETAILS and get PAYMENT.DATE, PROPERTY, PAYMENT.TYPE and OS.PROP.AMOUNT 2. Assign PayType variable with value “R” 3. If any of PAYMENT.TYPE is equal to “ACTUAL” then assign PayType variable with value “D” 4. If any of PAYMENT.TYPE is like to “PAYOFF” then assign PayType variable with value “P” 5. Using Loop statement extract respective PROPERTY & OS.PROP.AMOUNT  * Form the value as shown below and add it to the List array   UnpaidArrList = ProdGrp+”\*”+ PAYMENT.DATE+”\*”+ PROPERTY+”\*”+ ArrangementIdVal+”\*”+ OS.PROP.AMOUNT+”\*”+ PayType  **FinalArrayFormation:**   * If arrangement id is exists in ArrangementIds array (Outgoing argument) then add the extracted amount to the respective AllocAmount in the array.   Example:  extracted amount is = 50  Arrangement id = AA20001  Existing Array:   |  |  |  | | --- | --- | --- | | ArrangementIds | AllocAmount | PayTypes | | AA20001 | 2000 | R | | AA20002 | 40 | R |   After Update:   |  |  |  | | --- | --- | --- | | ArrangementIds | AllocAmount | PayTypes | | AA20001 | 2050 | R | | AA20002 | 40 | R |  * If arrangement id is not exists in ArrangementIds array (Outgoing argument) then add the new row in ArrangementIds, AllocAmount, PayTypes to store arrangement id, amount and paytype respectively.   Example:  extracted amount is = 75  Arrangement id = AA20008  Paytype = R  Existing Array:   |  |  |  |  | | --- | --- | --- | --- | | ArrangementIds | AllocAmount | PayTypes | Unc | | AA20001 | 2000 | R |  | | AA20002 | 40 | R |  |   After Update:   |  |  |  |  | | --- | --- | --- | --- | | ArrangementIds | AllocAmount | PayTypes | Unc | | AA20001 | 2000 | R |  | | AA20002 | 40 | R |  | | AA20008 | 75 | R |  | |
| Special Instructions |  |

## HusInvoiceReversalLogic

|  |  |
| --- | --- |
| Property | Specification |
| Type | S |
| Attached To | AA.PRD.DES.ACTIVITY.API>MORTGAGE--20100101  Activity Class: LENDING-ISSUEBILL-PAYMENT.SCHEDULE  Property Class: PAYMENT.SCHEDULE  Action: ISSUE.BILL  Activity Class: LENDING-ISSUEBILL-CHARGE  Property Class: CHARGE  Action: ISSUE.BILL  Activity Class: LENDING-ISSUEBILL-PERIODIC.CHARGES  Property Class: PERIODIC.CHARGES  Action: ISSUE.BILL  Activity Class: LENDING-CAPTURE.BILL-BALANCE.MAINTENANCE  Property Class: PAYMENT.SCHEDULE  Action: UPDATE.SCHEDULES  Activity Class: LENDING-ADJUST.BILL-BALANCE.MAINTENANCE  Property Class: PAYMENT.SCHEDULE  Action: UPDATE.SCHEDULES  Activity Class: LENDING-ADJUST.INFO.BILL-BALANCE.MAINTENANCE  Property Class: BALANCE.MAINTENANCE  Action: DATA.CAPTURE |
| Attached As | Post Routine |
| Dependency | NA |
| Description | Post routine will get reversed/deleted bill details and update EB.HB.INVOICE.DETAILS |
| Arguments – IN | NA |
| Arguments – OUT | NA |
| Prelim Conditions |  |
| Subroutine Flow | 1. Get activity status and if it is “AUTH-REV” then continue further else exit the routine. 2. Get account id using getLinkedAccount from arrangementContext and store it to the variable AccountId 3. Get arrangement id using getArrangementId from arrangementContext and store it to the variable ArrangementId 4. Get activity effective date using getActivityEffectiveDate from arrangementContext and store it to the variable ActivityDate 5. Read ACCOUNT record with @ID as AccountId and get HB.CASE.ID and store it to the variable CaseId 6. Read EB.HB.UNPAID.INVOICE record with @ID as Case Id and get INVOICE.DET.ID all multi-value 7. Using Loop statement, extract each INVOICE.DET.ID and store it the variable InvoiceDetid and do the below steps  * Using split, extract Invoice date from InvoiceDetid and store the result to Invdate i.e extract third value delimited by dot (.)   Example: InvoiceDetid = I.23412.20210101.NOK  Invdate = 20210101   * If Invdate is equal to ActivityDate then continue further else take next value from loop. * Read EB.HB.INVOICE.DETAILS record with @ID as InvoiceDetid * Locate ArrangementId in EB.HB.INVOICE.DETAILS> ARRANGEMENT.ID and get respective multi-value of BILL.ID * Remove each BILL.ID and add it to InvoiceDetid with \* Delimiter and store the result to array variable OlderBills   Example: OlderBills = AABILL20001\*I.23412.20210101.NOK   1. Using getBillIdsForDate API, get all Bill id’s created for the date in ActivityDate and store it to the array variable CurrentBills 2. Using Loop statement, extract each value from OlderBills array and do the below steps  * Using split, extract the value before \* from OlderBills and store it the variable OldbillIdval and store the value after \* to InvDetIdval * Check if OldbillIdval is existing in CurrentBills array. If it is not existing then add the respective OlderBills value to DeletedBills array  1. Using Loop statement, extract each value from DeletedBills array and do the below steps  * Using split, extract the value before \* from respective DeletedBills and store it the variable BillId and store the value after \* to InDetIdval * Read EB.HB.INVOICE.DETAILS table record with @ID as InDetIdval and get STATUS field value * If EB.HB.INVOICE.DETAILS>STATUS is equal to “NEW” or SENDING-GET-CHANNEL or FAILED-GET-CHANNEL then   + Locate ArrangementId in EB.HB.INVOICE.DETAILS>ARRANGEMENT.ID and within that respective associated multi-value locate BillId in EB.HB.INVOICE.DETAILS>BILL.ID   + Remove that respective BILL.ID multi-value and update TOTAL.AMOUNT * If EB.HB.INVOICE.DETAILS>STATUS is not equal to “NEW” then   + Update EB.HB.INVOICE.DETAILS>TYPE with “CREDIT.NOTE” and STATUS with “NEW” |
| Special Instructions |  |

## HusBatInvoicePreparation

|  |  |
| --- | --- |
| Property | Specification |
| Type | S |
| Attached To | BATCH>BNK/HUS.B.INVOICE.PREPARATION |
| Attached As | Batch Routine |
| Dependency | NA |
| Description | This routine used to update EB.HB.INVOICE.DETAILS with bills details which are created in COB |
| Arguments – IN | NA |
| Arguments – OUT | NA |
| Prelim Conditions |  |
| Subroutine Flow | **getIds:**   1. Select all records of EB.HB.CASE.TODAY.BILLS   **postUpdateRequest:**   1. Assign incoming id to the variable CaseIdval 2. Read EB.HB.CASE.TODAY.BILLS record with @ID as CaseIdval 3. Using Loop statement, remove each value from ARRANGEMENT.ID multi-value and do the below steps  * Call getBillIdsForDate with billDate argument equal to Today date to get today generated bills. * Loop through all the bills and do the steps explained in **ProcessBillDetails** paragraph  1. Delete record from EB.HB.CASE.TODAY.BILLS for the id stored in CaseIdval   **ProcessBillDetails:**   1. Read AA.BILL.DETAILS record and get PAYMENT.INDICATOR, PROPERTY, PAYMENT.TYPE, BILL.DATE and CURRENCY field value 2. Check PAYMENT.INDICATOR, PROPERTY and PAYMENT.TYPE and assign value to IdType   If PAYMENT.INDICATOR EQ “DEBIT” AND (PAYMENT.TYPE EQ “PAYOFF$CURRENT” OR PAYMENT.TYPE EQ “SPECIAL” OR PAYMENT.TYPE EQ “LOSS.SHARING” OR PROPERTY EQ “PREMIUM”(Any of the multi-value of PROPERTY)) then  IdType = “P”  If PAYMENT.INDICATOR EQ “CREDIT” then  IdType = “C”  If IdType is equal to empty then  IdType = “I”     1. Extract first multi-value from AA.BILL.DETAILS>BILL.DATE and store it to the variable InvoiceDate 2. Frame EB.HB.INVOICE.DETAILS record id as below   InvoiceDetId = IdType +”.”+ CaseIdVal+”.”+ InvoiceDate+”.”+ AA.BILL.DETAILS> CURRENCY   1. Read EB.HB.INVOICE.DETAILS record with @ID as InvoiceDetId 2. If record not found in EB.HB.INVOICE.DETAILS, form the OFS message using “TransactionData” argument to create new record with below field mapping.   TransactionId = InvoiceDetId  Version = EB.HB.INVOICE.DETAILS,HUS.INPUT  Function = “INPUT”   |  |  |  | | --- | --- | --- | | **Field name** | **MV** | **Mapping** | | CASE.ID |  | AA.ARR.ACCOUNT>HUS.CASE.ID | | INITIAL.DUE.DATE |  | AA.BILL.DETAILS>PAYMENT.DATE | | CURRENCY |  | AA.BILL.DETAILS>CURRENCY | | CUSTOMER.ID | XX< | EB.HUS.LA.CASE>PROJECT.PARTY where SEND.INVOICE is not equal to NO  If AA.BILL.DETAILS>PAYMENT.TYPE is equal to “FINAL.DEMAND.PAYMENT” add all values in PARTY.ROLE irrespective of SEND.INVOICE field value. | | REGISTER.ID | XX- | CUSTOMER>HUS.REGISTER.ID | | DBT.REST.STATUS | XX> | CUSTOMER>DBT.REST.STATUS | | CUSTOMER.TYPE |  | Read CUSTOMER record and get CUSTOMER.STATUS field value.  Check CUSTOMER.STATUS field value existing in EB.HB.INVOICE.PARAM> B2B.CUSTOMER.STATUS if found then default this field with value B2B  Check CUSTOMER.STATUS field value existing in EB.HB.INVOICE.PARAM>B2C.CUSTOMER.STATUS if found then default this field with value B2C | | CUSTOMER.REF |  | EB.HUS.LA.CASE>CUS.INVOICE.REF | | ARRANGEMENT.ID | XX< | AA.BILL.DETAILS>ARRANGEMENT.ID | | LINE.ID | XX-XX< | Sequential number starting from 1. | | AMOUNT.TYPE | XX-XX- | Locate AA.BILL.DETAILS> PROPERTY within EB.HB.INVOICE.PARAM> ACCOUNT.PROPERTY, PRINCIPAL.INT.PROPERTY, PENALTY.INT.PROPERTY, CHARGE.PROPERTY, COST.PROPERT, if found get the corresponding associated description field value from ACCOUNT.DESC, PRINCIPAL.INT.DESC, PENALTY.INT.DESC, CHARGE.DESC and COST.DESC | | AMOUNT | XX-XX- | AA.BILL.DETAILS> OR.PROP.AMOUNT | | BILL.ID | XX>XX> | AA.BILL.DETAILS>@ID | | TOTAL.AMOUNT |  | Sum of all AMOUNT fields | | ISSUE.DATE |  | TODAY Date | | STATUS |  | SENDING-GET-CHANNEL | | TYPE |  | INVOICE, if the id starts with I  “PAYMENT.REQUEST.PREMIUM” if AA.BILL.DETAILS>PROPERTY is equal to PREMIUM and id starts with “P”  “PAYMENT.REQUEST.PAYEXTRA” if EB.HUS.LA.CASE>IN.FLAG NE "Y" and id starts with “P”  “PAYMENT.REQUEST.PAYINDEBT” if CUSTOMER>TARGET EQ 1 and EB.HUS.LA.CASE>IN.FLAG EQ "Y" and id starts with “P”  “PAYMENT.REQUEST.PAYOFF” if PAYMENT.TYPE is "PAYOFF$CURRENT" and PRODUCT.GROUP NE "HUS.GRANTS" and id starts with “P”  “PAYMENT.REQUEST.PAYOFFGRANT” if PAYMENT.TYPE is "PAYOFF$CURRENT" and PRODUCT.GROUP EQ "HUS.GRANTS" and id starts with “P”  PAYMENT.REQUEST.REPAYGRANT if PAYMENT.TYPE is "SPECIAL" and PRODUCT.GROUP EQ "HUS.GRANTS" and id starts with “P”  CREDIT.NOTE, if the id starts with C | | DUE.DATE |  | AA.BILL.DETAILS>PAYMENT.DATE |  1. If record found, form the OFS message using “TransactionData” argument to update existing record with below field mapping.   TransactionId = InvoiceDetId  Version = EB.HB.INVOICE.DETAILS,HUS.OFS  Function = “INPUT”   |  |  |  | | --- | --- | --- | | **Field name** | **MV** | **Mapping** | | ARRANGEMENT.ID | XX< | If AA.BILL.DETAILS>ARRANGEMENT.ID isn’t present yet, create a new multi-value otherwise use the existing multi-value | | LINE.ID | XX-XX< | Take value from previous multi-value of LINE.ID and increment it if it is new multi-value. | | AMOUNT.TYPE | XX-XX- | Locate AA.BILL.DETAILS> PROPERTY within EB.HB.INVOICE.PARAM> ACCOUNT.PROPERTY, PRINCIPAL.INT.PROPERTY, PENALTY.INT.PROPERTY, CHARGE.PROPERTY, COST.PROPERT, if found get the corresponding associated description field value from ACCOUNT.DESC, PRINCIPAL.INT.DESC, PENALTY.INT.DESC, CHARGE.DESC and COST.DESC | | AMOUNT | XX-XX- | AA.BILL.DETAILS> OR.PROP.AMOUNT | | BILL.ID | XX>XX> | AA.BILL.DETAILS>@ID | | TOTAL.AMOUNT |  | Sum of all AMOUNT fields | |
| Special Instructions |  |

## HusPayoffSimRepayVal

|  |  |
| --- | --- |
| Property | Specification |
| Type | S |
| Attached To | AA.SIMULATION.RUNNER,HUS.AA |
| Attached As | Input routine |
| Dependency | NA |
| Description | This routine will raise an error/override message when extra-ordinary or individual payment initiated on fixed rated loan |
| Arguments – IN | NA |
| Arguments – OUT | NA |
| Prelim Conditions |  |
| Subroutine Flow | 1. Get ARRANGEMENT.REF and SIM.RUN.DATE field value from currentRecord and store it to the variable simArrId and actEffDate 2. Get AA.ARR.INTEREST>PERIODIC.INDEX and FLOATING.INDEX field value for an arrangement simArrId using getConditionForPropertyEffectiveDate method and store it the variable perIntIndex and floatIndex 3. Read AA.ARRANGEMENT record with @ID as simArrId and get CUSTOMER field value and store it to the variable custmerId 4. Read CUSTOMER record with @ID as custmerId and get TARGET field value and store it to the variable cusTarget 5. Get AA.ARR.ACCOUNT>HB.CASE.ID field value for an arrangement simArrId using getAccountCondition method and store it the variable caseId 6. Read EB.HUS.LA.CASE record with @ID as caseId and get IN.FLAG field value and store it to the variable inFlagVal 7. If inFlagVal is equal to “YES” and perIntIndex is not equal to null then raise an error message AA-HUS.CASE.IS.MARKED 8. If inFlagVal is equal to “YES” and floatIndex is not equal to null and cusTarget is equal to “1” then raise an override message AA-HUS.CASE.IS.MARKED 9. Update AA.SIMULATION.RUNNER>SIM.END.DATE field with value of AA.SIMULATION.RUNNER>SIM.RUN.DATE |
| Special Instructions |  |

## HusAaValInstallFreqCheck

|  |  |
| --- | --- |
| Property | Specification |
| Type | S |
| Attached To | AA.PRD.DES.ACTIVITY.API>MORTGAGE--20100101  Activity Class: LENDING-CHANGE-PAYMENT.SCHEDULE  Property Class: PAYMENT.SCHEDULE  Action: UPDATE |
| Attached As | Pre Routine |
| Dependency | NA |
| Description | Routine will validate the frequency change and will raise an error message if customer is not allowed to have that frequency. |
| Arguments – IN | NA |
| Arguments – OUT | NA |
| Prelim Conditions |  |
| Subroutine Flow | 1. Using getCustomer method from arrangementActivityRecord, get customer id and store it the variable customerId 2. Read CUSTOMER record with @ID as customerId and get TARGET field value and store it to the variable cusTarget 3. Read EB.HUS.GENERIC.PARAM with @ID as “ALLOWED.INSTALL.FQC” 4. Get EB.HUS.GENERIC.PARAM>PARAM.TYPE, loop through all multi-values and do the below steps  * If cusTarget value is existing in PARAM.TYPE (values are delimited by , (comma)) then get corresponding EB.HUS.GENERIC.PARAM> PARAM.NAME & PARAM.VALUE field values * Loop through each PARAM.NAME & PARAM.VALUE field values   + If PARAM.NAME equal to “MONTH.FQC” then get corresponding field value of PARAM.VALUE and store it the variable AllowedMntFqc  1. Get value from AA.ARR.PAYMENT.SCHEDULE>PAYMENT.FREQ for CONSTANT or LINEAR payment type. Using substring extract string between “e” and “M” from PAYMENT.FREQ. (Example: PAYMENT.FREQ = e0Y e1M e0W o1D e0F   Extracted value should be 1)   * If extracted value is not matching with any of the values stored AllowedMntFqc (values are delimited by , (comma)) then raise an error message AA-HUS.ALLOWED.INSTAL.FQC with extracted value as argument |
| Special Instructions |  |

## HusAuthPayOffTypeInvUpd

|  |  |
| --- | --- |
| Property | Specification |
| Type | S |
| Attached To | VERSION>EB.HUS.CUS.APP.LA.DOCU,OFS  Note: Table & Version developed as part of CR19 development |
| Attached As | Authorisation Routine |
| Dependency | NA |
| Description | This routine will update EB.HB.INVOICE.DETAILS record when authoriser approves the application. |
| Arguments – IN | NA |
| Arguments – OUT | NA |
| Prelim Conditions | NA |
| Subroutine Flow | 1. If currentRecord>CHANGE.TYPE is equal to “PRINC.DECREASE” then proceed further else exit the routine 2. Form the OFS message using “TransactionData” argument to create new record with below field mapping.   TransactionId = InvoiceDetId  Version = “EB.HB.INVOICE.DETAILS,HUS.PAYOFF”  Function = “INPUT”  OFS Source = “PAYOFF.OFS”   |  |  |  | | --- | --- | --- | | **Field name** | **MV** | **Mapping** | | CASE.ID |  | currentRecord>CASE.ID | | INITIAL.DUE.DATE |  | currentRecord> SIMULATION.REFERENCE >>  AA.SIMULATION.RUNNER>SIM.RUN.DATE | | CURRENCY |  | currentRecord>ARR.ID  AA.ARRANGEMENT>CURRENCY | | CUSTOMER.ID | XX< | currentRecord>CUS.ID | | REGISTER.ID | XX- | currentRecord> CUS.ID **>>** CUSTOMER>HUS.REGISTER.ID | | DBT.REST.STATUS | XX> | CUSTOMER>DBT.REST.STATUS | | CUSTOMER.TYPE |  | Read CUSTOMER record and get CUSTOMER.STATUS field value.  Check CUSTOMER.STATUS field value existing in EB.HB.INVOICE.PARAM> B2B.CUSTOMER.STATUS if found then default this field with value B2B  Check CUSTOMER.STATUS field value existing in EB.HB.INVOICE.PARAM>B2C.CUSTOMER.STATUS if found then default this field with value B2C | | CUSTOMER.REF |  | currentRecord>CASE.ID **>>**  EB.HUS.LA.CASE>CUS.INVOICE.REF | | ARRANGEMENT.ID | XX< | currentRecord>ARR.ID | | LINE.ID | XX-XX< | “1” | | AMOUNT.TYPE | XX-XX- | “Principal” | | AMOUNT | XX-XX- | currentRecord> SIMULATION.REFERENCE >>  AA.SIMULATION.RUNNER>T.OVR.AMOUNT | | BILL.ID | XX>XX> | currentRecord>SIMULATION.REFERENCE | | TOTAL.AMOUNT |  | currentRecord> SIMULATION.REFERENCE >>  AA.SIMULATION.RUNNER>T.OVR.AMOUNT | | ISSUE.DATE |  | TODAY Date | | STATUS |  | “NEW” | | TYPE |  | currentRecord> CUS.ID **>>** CUSTOMER>TARGET  If CUSTOMER>TARGET is equal to 1 then default “PAYMENT.REQUEST.PAYINDEBT” else default this field with value “PAYMENT.REQUEST.PAYEXTRA” | | DUE.DATE |  | currentRecord> SIMULATION.REFERENCE >>  AA.SIMULATION.RUNNER>SIM.RUN.DATE | | PREMIUM.INFO |  | * Form account details record id as show below   accDetConId = currentRecord>ARR.ID+"%"+currentRecord> SIMULATION.REFERENCE   * Read AA.ACCOUNT.DETAILS$SIM record with @ID as accDetConId * Get latest bill id from AA.ACCOUNT.DETAILS$SIM>BILL.ID and store it the variable currBillId * Form bill details record id as shown below   simBillId = currBillId +"%"+ currentRecord> SIMULATION.REFERENCE   * Read AA.BILL.DETAILS$SIM record with @ID as simBillId and get OR.PROP.AMT for PREMIUM property and default that value to this field | | DISCOUNT.INFO |  | * Form account details record id as show below   accDetConId = currentRecord>ARR.ID+"%"+currentRecord> SIMULATION.REFERENCE   * Read AA.ACCOUNT.DETAILS$SIM record with @ID as accDetConId * Get latest bill id from AA.ACCOUNT.DETAILS$SIM>BILL.ID and store it the variable currBillId * Form bill details record id as shown below   simBillId = currBillId +"%"+ currentRecord> SIMULATION.REFERENCE   * Read AA.BILL.DETAILS$SIM record with @ID as simBillId and get OR.PROP.AMT amount for DISCOUNT property and default that value to this field | |
| Special Instructions |  |

## HusAuthPayOffTypeStaging

|  |  |
| --- | --- |
| Property | Specification |
| Type | S |
| Attached To | VERSION>EB.HB.INVOICE.DETAILS,HUS.PAYOFF |
| Attached As | Authorisation Routine |
| Dependency | NA |
| Description | This routine will update EB.HB.INVOICE.STAGING record when authoriser approves the application. |
| Arguments – IN | NA |
| Arguments – OUT | NA |
| Prelim Conditions | NA |
| Subroutine Flow | 1. Form the OFS message using “TransactionData” argument to create new record in EB.HB.INVOICE.STAGING with below field mapping.   TransactionId = currentRecord>KID.NUMBER  Version = EB.HB.INVOICE.STAGING,HUS.PAYOFF  Function = “INPUT”  OFS Source = “PAYOFF.OFS” |
| Special Instructions |  |

# Appendix

NA

# Accounting

NA

# Limits

NA

# COB Processing

NA

# Assumptions

1. The case reference would be generated during the loan application process and provided to the AA module during the loan opening.
2. The only currency used on loans and grants is NOK
3. Delivered sub-menu will be attached to main menu in SMS development.

# Dependencies/Exclusions

1. This requirement is dependent on the following solutions:

* FSD\_LA-Interim Workflows
* FSD\_LM01 - Data model - Loan Maintenance
* ISD\_#011.1.Billing and Reminder
* TSD\_BP04\_ Transfer of Loans
* TSD\_CR19Part2A\_LoanAmendment

# Testing Notes

Please find DIT in the attachment



# Packaging / Installation

NA

# Data Migration

NA