Oracle® Banking Platform Collections

Interface Specification Guide

Release 2.4.1.0.0

E70795-01

February 2016

Oracle Banking Platform Collections Interface Specification Guide, Release 2.4.1.0.0

E70795-01

Contents

List of Tables

Preface

This document covers the staging data table structure and the services exposed by the system for host systems to use.

This preface contains the following topics:

Audience

This document is intended for the following audience:

- IT Deployment Team
- Consulting Staff
- Administrators

Documentation Accessibility

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

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Organization of the Guide

This document contains:

Chapter 1, "Introduction"

This chapter presents an overview of staging area and services exposed.

Chapter 2, "System Overview"

This chapter provides information about the modules or systems interfaced with OBP Collections.

Chapter 3, "Staging Area"

This chapter provides details of the feeder tables.

Chapter 4, "Algorithms"

This chapter outlines the pre-shipped algorithm details.

Chapter 5, "Feeder Services"

This chapter lists the services exposed by collections for data updates.

Related Documents

For more information, see the following documentation:

- For the complete list of the adapters for integration with Oracle Banking Platform modules and technology stacks such as DMS / Alert /Email systems, see the Oracle Banking Platform Collections Adapter Configuration Guide.

Conventions

The following text conventions are used in this document:

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

1 Introduction

In Oracle Banking Platform, Collection system identifies delinquent accounts, fetches the account and party related data and stores it in the staging tables. After validation of these records, entity creation batch processes these records and moves them to Collections tables. For other host systems, it is expected that delinquent account data is pushed into these staging tables.

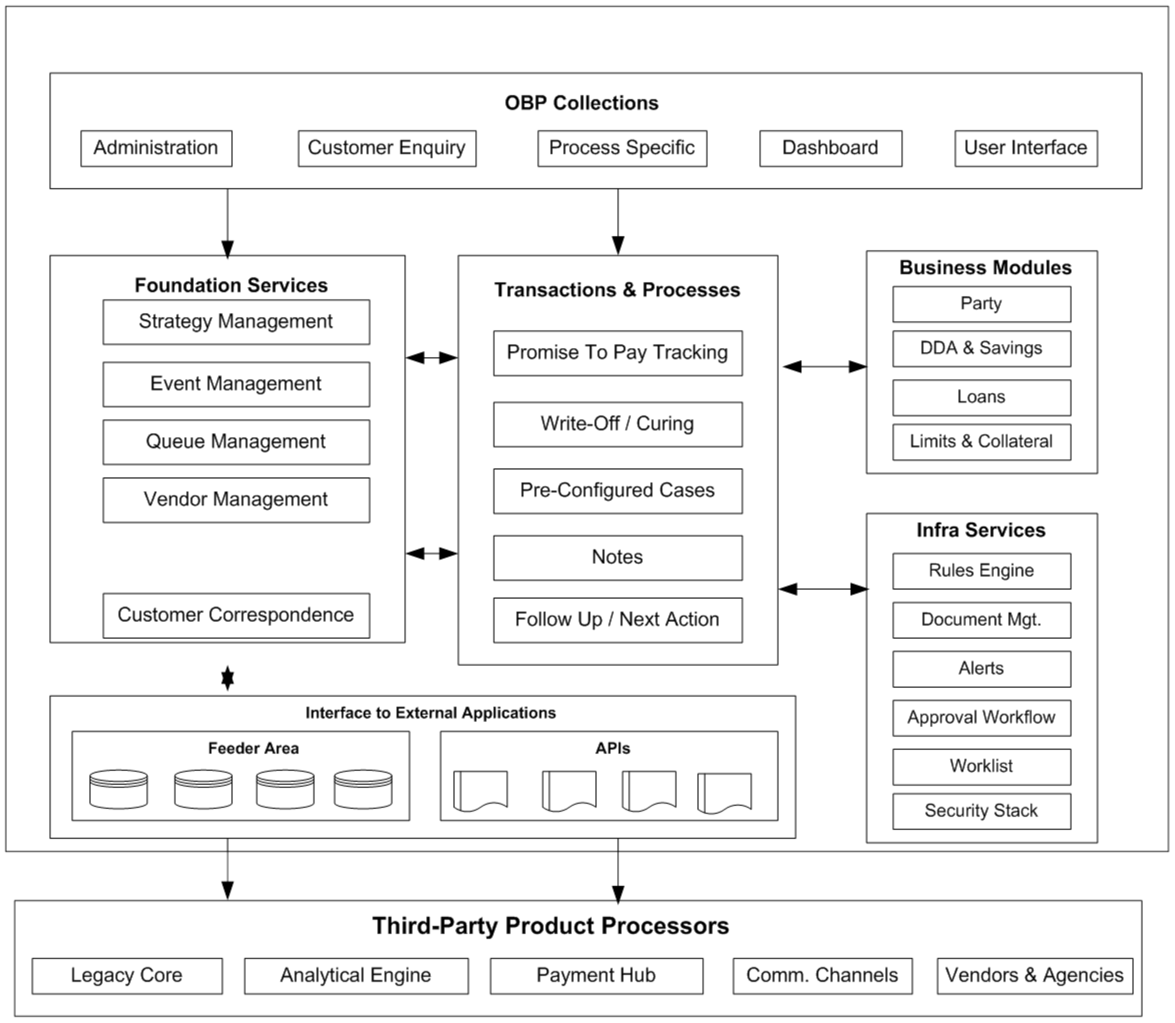
The feeder services exposed by Collections are invoked when changes in data take place in OBP. These services bring modified data into staging tables before batch processes these and update collections tables.

2 System Overview

This chapter provides information about the modules or systems interfaced with OBP Collections.

The diagram below shows the interface that Collections has with other modules or systems. It depicts the collections flow and its interface with OBP modules.

Figure 2–1 System Overview



3 Staging Area

This chapter provides information about the modules or systems interfaced with OBP Collections.

3.1 Feeder Tables

The feeder tables listed in this section provide a staging area for the host systems to push data. Offline collection batch process reads this data and creates accounts in Collections.

3.1.1 Account Data

This section provides information on the tables related to accounts.

3.1.1.1 Account Details

Table Name: Account Details (CI_FDR_ACCT)

Description: This table holds account related data from host.

Table 3–1 Account Details

Field Name	Description	Values	Data Type	Length	Required	Column Name
Account No	Account Number as stored in Host		VARCHAR 2	40	Y	HOST_ACCT_ NBR
Host ID	Source Host ID for host		VARCHAR 2	10	Y	SRC_HOST_ID
Business Unit	Business Unit of the Account. This field is used only if multi-branding features are to be used.		VARCHAR 2	40	N	BUSINESS_ UNIT
Market Entity	Market Entity to which account belongs. This field is used only if multi-branding features are to be used.		VARCHAR 2	40	N	MARKET_ ENTITY
Facility ID	Facility ID under which account is created. This field is used based on the structure of accounts in the host.		VARCHAR 2	40	N	FACILITY_ID

Table 3–1 (Cont.) Account Details

Field Name	Description	Values	Data Type	Length	Required	Column Name
Liability ID	Liability ID under which the Facility ID of the account has been created. This field is used based on the structure of accounts in the host.		VARCHAR 2	40	N	LIABILITY_ID
Product Class	Product Class of the account	Lending, CASA	VARCHAR 2	10	Y	HOST_PROD_ CLASS_CD
Product Group	Product Group associated with the account	Auto, Loan, and so on	VARCHAR 2	30	Y	HOST_PROD_ GRP_CD
Product Code	Code of the banking product offered to the customer		VARCHAR 2	10	Y	HOST_PROD_ CD
System Account Status	As defined in the host	Regular, Dormant, Closed, Written Off	VARCHAR 2	20	Y	HOST_SYS_ ACCT_STAT_ FLG
User defined Account Status	As defined in the host	For example, Debit Block, Credit Block, and so on.	VARCHAR 2	100	N	USR_DEF_ ACCT_STAT_ FLG
Accrual Status	This field displays the accrual status for the account.	Normal, Suspended	CHAR	1	Y	ACCRL_STAT_ FLG
Asset Classification Code	As identified by the host		VARCHAR 2	30	Y	ASST_CLASS_ CD
Repayment Frequency	Repayment Frequency of the loan	Monthly, weekly, quarterly	VARCHAR 2	30	N	REPAYMNT_ FREQ
Un-Cleared Payment Amount	Sum of all uncleared credits to the account		NUMBER	36,18	N	UNCLR_ PAYMNT_AMT
Loan Maturity Date	Date when loan matures		DATE	10	Y	MATURITY_ DT
Redraw Count	Number of times a redraw has been performed		NUMBER	3,0	N	REDRAW_CNT
Account Write Off Date	Date when account is fully written off/ abandoned		DATE	10	N	WRITE_OFF_ DT
Account Write Off Amount	Written off loan amt (abandonment amount). Total of all sums written off will be given.		NUMBER	36,18	N	WRITE_OFF_ AMT
Last Provision Date	Date on which the provision entry was last accounted		DATE	10	N	LAST_ PROVSN_DT

Table 3–1 (Cont.) Account Details

Field Name	Description	Values	Data Type	Length	Required	Column Name
Provision Balance	Latest balance in Provision GL for the account		NUMBER	36,18	N	LAST_ PROVSN_BAL
Last Principal Write Off date	Date on which the principal write off entry was last passed		DATE	10	N	LAST_ PRNCPL_ WRITE_OFF_ DT
Principal Write Off Balance	Latest balance in Principal Write Off GL for the account		NUMBER	36,18	N	LAST_ PRNCPL_ WRITE_OFF_ BAL
Loan Purpose Type	Loan purpose types as applicable to the host		VARCHAR 2	20	N	ACCT_PURPS_ TYPE
Loan Purpose Code	List of values as per loan purpose type		VARCHAR 2	20	N	ACCT_PURPS_ CD
Date of last loan restructure	Date when the loan was last restructured		DATE	10	N	LAST_ACCT_ RESTR_DT
Offer ID	Offer ID applicable to the customer account		VARCHAR 2	30	N	OFFER_ID
Offer Name	Offer Name as per the Offer ID provided		VARCHAR 2	60	N	OFFER_NAME
Account Opening Date or Initial Disbursement Date	Term Loan: First Disbursement Date OD: Date on which OD facility is provided Current Account with TOD facility: TOD utilization Date - Derived		DATE	10	Y	SETUP_DT
Account Currency Code	Currency code of the account		VARCHAR 2	3	Y	ACCT_CURR_ CD
Outstanding Amount	Outstanding Amount for the account	**OD Accounts**: OD Limit Utilized + AUF Limit Utilized + Overdue Amount **Term Loans** : Outstanding Principal - RPA Balance + Overdue Amount	NUMBER	36,18	Y	OUTSTANDIN G_AMT

Table 3–1 (Cont.) Account Details

Field Name	Description	Values	Data Type	Length	Required	Column Name
Overdue Amount	Overdue amount for the account	**OD Accounts**: TOD utilized + Overline utilized + Temporary Excess utilized **Term Loans :** All amounts due and still unpaid	NUMBER	36,18	Y	OVERDUE_ AMT
Account Limit	Sanctioned Limit offered to the account	**OD Accounts** : OD limit + Temporary Excess limit **Term Loans :** Sanctioned Amount	NUMBER	36,18	Y	OVERLIMIT_ AMT
DPD	Longest Days past due value computed by the host		NUMBER	4,0	Y	DAYS_PAST_ DUE
Delinquency Start Date	Current Delinquency Start Date. To be sent only once with the initial data hand off.		DATE	10	N	DEL_START_ DT
Installment(s) in Arrears	Total number of installments in arrears	Installment amount can at most consist of Principal, Interest and Fees. Even if one of the components is not fully paid; the installment will be construed as 'In Arrears'.	NUMBER	4,0	N	INSTALLMEN T_IN_ARS
Disbursed Amount	Amount disbursed so far in case of a tranche		NUMBER	36,18	N	DISBRS_AMT
Available for Disbursement	Total loan amount available for disbursement		NUMBER	36,18	N	TOTL_AVL_ DISBRS_AMT
Last Payment Date	Last Payment Date - Customer initiated credit.		DATE	10	N	LAST_ PAYMENT_DT
Last Payment Amount	Last Payment Amount - Customer initiated credit.		NUMBER	36,18	N	LAST_ PAYMENT_ AMT
Amount of Debit Interest Accrued	Applicable only to accounts with Debit balance		NUMBER	36,18	N	DR_INT_ ACCRD_AMT
Interest Rate	Rate of interest for current applicable stage		NUMBER	5,0	Y	INT_RATE

Table 3–1 (Cont.) Account Details

Field Name	Description	Values	Data Type	Length	Required	Column Name
Interest Type	Fixed or Floating		VARCHAR 2	14	Y	INT_TYPE
Address Type Code	Overriding address type configured for an account		VARCHAR 2	20	N	ADDR_TYPE_CD
Employee Account Flag	Indicate if the account belongs to a bank employee	Y/N	VARCHAR 2	1	Y	EMPLOYEE_ACCT_FLG
Minor Account Status	Indicate if the account belongs to a minor	Y/N	VARCHAR 2	40	Y	MINOR_ACCOUNT_STATUS_TYPE
Home Branch	Home Branch of the account		VARCHAR 2	20	Y	BRANCH_CD
User Defined Field 1	User Defined Field in case any additional attributes are required	**Exposure at Default :** String value coming from third party interface	VARCHAR 2	60	N	UDF1
User Defined Field 2	User Defined Field in case any additional attributes are required	**Loss Given Default :** String value coming from third party interface	VARCHAR 2	60	N	UDF2
User Defined Field 3	User Defined Field in case any additional attributes are required	**Expected Loss :** String value coming from third party interface	VARCHAR 2	60	N	UDF3
User Defined Field 4	User Defined Field in case any additional attributes are required	**Risk Weighted Asset Calculation :** String value coming from third party interface	VARCHAR 2	60	N	UDF4
User Defined Field 5	User Defined Field in case any additional attributes are required		VARCHAR 2	60	N	UDF5
User Defined Field 6	User Defined Field in case any additional attributes are required		VARCHAR 2	60	N	UDF6
User Defined Field 7	User Defined Field in case any additional attributes are required		VARCHAR 2	60	N	UDF7
User Defined Field 8	User Defined Field in case any additional attributes are required		VARCHAR 2	60	N	UDF8
User Defined Field 9	User Defined Field in case any additional attributes are required		VARCHAR 2	60	N	UDF9

Table 3–1 (Cont.) Account Details

Field Name	Description	Values	Data Type	Length	Required	Column Name
User Defined Field 10	User Defined Field in case any additional attributes are required		VARCHAR 2	60	N	UDF10
User Defined Field 11	User Defined Field in case any additional attributes are required		VARCHAR 2	60	N	UDF11
User Defined Field 12	User Defined Field in case any additional attributes are required		VARCHAR 2	60	N	UDF12
User Defined Field 13	User Defined Field in case any additional attributes are required		VARCHAR 2	60	N	UDF13
User Defined Field 14	User Defined Field in case any additional attributes are required		VARCHAR 2	60	N	UDF14
User Defined Field 15	User Defined Field in case any additional attributes are required		VARCHAR 2	60	N	UDF15
User Defined Field 16	User Defined Field in case any additional attributes are required		VARCHAR 2	60	N	UDF16
User Defined Field 17	User Defined Field in case any additional attributes are required		VARCHAR 2	60	N	UDF17
User Defined Field 18	User Defined Field in case any additional attributes are required		VARCHAR 2	60	N	UDF18
User Defined Field 19	User Defined Field in case any additional attributes are required		VARCHAR 2	60	N	UDF19
User Defined Field 20	User Defined Field in case any additional attributes are required		VARCHAR 2	60	N	UDF20
Reason for Delinquency	Reason code for delinquency of the account		VARCHAR 2	40	N	HOST_ REASON_ FOR_ DELINQUENC Y
Redraw Availability	Facility to redraw loan	Y/N	CHAR	1	Y	FDR_ REDRAW_ AVL_SW
Joint Applicant	Indicates if the account has a Joint Applicant	Y/N	VARCHAR 2	1	Y	FDR_JOINT_ APPLICANT_ SW
Delinquent	Indicates if the account is delinquent	Y/N	VARCHAR 2	1	Y	FDR_IS_ DELINQUENT _SW
Non Starter	Indicates if the customer defaults the first installment after disbursement	Y/N	VARCHAR 2	1	Y	FDR_NON_ STARTER_SW

Table 3–1 (Cont.) Account Details

Field Name	Description	Values	Data Type	Length	Required	Column Name
Behavior Score	Current Behavior Score captured at account level		VARCHAR 2	10	N	FDR_ BEHAVIOR_ SCORE
Probability of Default	Current Probability of default captured at account level		VARCHAR 2	60	N	PROBABILITY _OF_DEFLT_ VAL
Application Score	Application Score captured at the time of opening of account		VARCHAR 2	10	N	FDR_APPL_ SCR
Loan to Value Ratio	Loan to Value Ratio (Book/ Bank Value is considered) - Value of External Charge on Collateral is considered while calculating LVR		NUMBER	5,2	N	FDR_LTV_VAL
Loan to Value Ratio	Loan to Value Ratio (MTM is considered) - Value of External Charge on Collateral is considered while calculating LVR		NUMBER	5,2	N	FDR_LVR_VAL
Joint Nomination flag	Joint Nomination flag		VARCHAR 2	1	N	FDR_JOINT_ NOMINATION _SW
Record Type	Signifies if the data is created initially or is updated for existing data	I - Insert U - Update	VARCHAR 2	10	Y	RCD_TYPE
Record Creation Date	Date on which data is fed to Collections		DATE	10	Y	CRET_DTTM
BICOE loan account Switch	BICOE loan account Switch		CHAR	1	N	BICOE_LOAN_ SW
Customer Class Code	Customer Class Code		VARCHAR 2	8	N	CUST_CL_CD
First Default date	First Default date		DATE	10	N	FIRST_ DEFAULT_ DATE
Last Days Past Due update Date	Last Days Past Due		DATE	10	N	LAST_DPD_ UPDATE_DT
Relationship Officer Code	Relationship Officer Code		VARCHAR 2	40	N	RELATION_ OFFICER_ CODE
FDR_FORCED_ SW	FDR Forced SW		VARCHAR 2	1	Y	FDR_FORCED_ SW
FORCED_ REASON_CD	Forced Reason CD		VARCHAR 2	4	Y	FORCED_ REASON_CD
IOA_ BALANCE_ AMT	IOA Balance Amount		NUMBER	36,18	N	IOA_ BALANCE_ AMT

3.1.1.2 Account Arrears Details

Table Name: Account Arrear Details (CI_FDR_ACCT_ARS)

Description: This table holds account arrears data from host.

Table 3–2 Account Arrears Details

Field Name	Description	Values	Data Type	Length	Required	Column Name
Account No	Account Number as stored in Host		VARCHAR2	40	Y	HOST_ACCT_ NBR
Host ID	Source Host ID for host		VARCHAR2	10	Y	SRC_HOST_ID
Sequence Number	Sequence Number for arrear type		VARCHAR2	50	Y	REFERENCE_ VAL
Arrear Type	Arrear type like interest, fee, and so on		VARCHAR2	40	N	ARS_TYPE
Arrear Amount	Total arrear rose per arrear type. Details of arrear type should be sent only where arrear amount > 0		NUMBER	36,18	N	ARS_ ASSESSED_ AMT
Paid Amount	Amount paid so far. Zero if no payments are received.		NUMBER	36,18	N	ARS_PAID_ AMT
Arrear Due	As calculated by Host		NUMBER	36,18	N	ARS_DUE_ AMT
Last Payment Date	Date when last payment was received		DATE	10	N	LAST_ PAYMENT_DT
Days in Arrear	Days this arrear is open. Zero is a valid value.		NUMBER	4,0	N	DAYS_IN_ARS
Installment Number	Installment Number		NUMBER	5,0	N	INSTALLMEN T_NUM
Record Creation Date	Date on which data is fed to Collections.		DATE	10	Y	CRET_DTTM
Record Type	Signifies if the data is created initially or is updated for existing data	I - Insert U - Update	VARCHAR2	10	Y	RCD_TYPE
Process Status	To check the current status of process. Default is P-Pending.		VARCHAR2	1	Y	PROCESS_ STATUS
Message Category Number	Defined error message category		NUMBER	5,0	Y	MESSAGE_ CAT_NBR
Message Number	Error message number		NUMBER	5,0	Y	MESSAGE_ NBR
Record Update Date	Date on which the record is updated		DATE	10	N	RECORD_ UPDATE_DT
Record Exist Switch	To check whether record is already available or not		VARCHAR2	1	Y	RECORD_ EXISTS_SW
ARS_DUE_ DT	RES due date		DATE	10	N	ARS_DUE_DT

3.1.1.3 Account Hardship Details

Table Name: Account Hardship Details (CI_FDR_ACCT_HARDSHIP_DTLS)

Description: This table holds account hardship data from host.

Table 3–3 Account Hardship Details

Field Name	Description	Values	Data Type	Length	Required	Column Name
Account No	Account Number as stored in Host		VARCHAR2	40	Y	HOST_ACCT_NBR
Host ID	Source Host ID for host		VARCHAR2	10	Y	SRC_HOST_ID
Application ID	Hardship Application ID		VARCHAR2	40	Y	HARSHIP_APPLICATION_ID
Relief Effective Date	Will be unique per Application ID		DATE	10	Y	RELIEF_EFFECTIVE_DT
Relief Expiry Date	Will be unique per Application ID		DATE	10	Y	RELIEF_EXPIRY_DT
Relief Type(s)	Can be more than 1 per application ID		VARCHAR2	40	Y	RELIEF_TYPE
Number of Payments Waived	Number of Payments Waived		NUMBER	4,0	N	NO_PAYMNT_WAIVED
User's Discretionary Margin (UDM)	These field details will be received only in case of Change Interest Rate relief type.		VARCHAR2	60	N	USR_DISCRTN_MRGN
UDM Start Date	User's discretionary Margin start date for the relief		DATE	10	N	USR_DISCRTN_MRGN_START_DT
UDM End Date	User's discretionary Margin end date for the relief		DATE	10	N	USR_DISCRTN_MRGN_END_DT
Reason for UDM	Reason for User's discretionary Margin		VARCHAR2	200	N	USR_DISCRTN_MRGN_RSN
Status	Current Status of Hardship Relief if applicable		CHAR	60	N	STATUS
Original Relief Type	Original Relief Type		VARCHAR2	40	N	ORIG_RELIEF_TYPE
Record Creation Date	Date on which the data is fed to Collections		DATE	10	N	CRET_DTTM
Record Type	Signifies if the data is created initially or is update for existing data	I - Insert U - Update	VARCHAR2	10	Y	RCD_TYPE
Process Status	To check the current status of process. Default is P-Pending.		VARCHAR2	1	Y	PROCESS_STATUS

Table 3–3 (Cont.) Account Hardship Details

Field Name	Description	Values	Data Type	Length	Required	Column Name
Message Category Number	Defined error message category		NUMBER	5,0	Y	MESSAGE_CAT_NBR
Message Number	Error message number		NUMBER	5,0	Y	MESSAGE_NBR
Record Update Date	Date on which the record is updated		DATE	10	N	RECORD_UPDATE_DT
Record Exist Switch	To check whether record is already available or not		VARCHAR2	1	Y	RECORD_EXISTS_SW

3.1.1.4 Account Repayment Schedule

Table Name: Account Repayment Schedule (CI_FDR_REPAYMENT_SCH)

Description: This table holds account repayment schedule data from host.

Table 3–4 Account Repayment Schedule

Field Name	Description	Value	Data Type	Length	Required	Column Name
Account No	Account Number as stored in Host		VARCHAR2	40	Y	HOST_ACCT_NBR
Host ID	Source Host ID for host		VARCHAR2	10	Y	SRC_HOST_ID
Date	Date when the installments are to be recovered		DATE	10	Y	INSTALLMENT_DT
Amount	Installment amount		NUMBER	36,18	N	INSTALLMENT_AMT
Principal	Principal component		NUMBER	36,18	N	PRINCIPAL_AMT
Interest	Interest component		NUMBER	36,18	N	INTEREST_AMT
Fee	Fee component, if any		NUMBER	36,18	N	FEE_AMT
Balance	Outstanding balance after the installment is paid		NUMBER	36,18	N	PRINCIPAL_BALANCE
Installment Number	Installment number as per the loan structure		NUMBER	5,0	N	INSTALLMENT_NUM
Record Creation Date	Date on which data is fed to Collections		DATE	10	N	CRET_DTTM
Record Type	Signifies if the data is created initially or is update for existing data	I - Insert U - Update	VARCHAR2	10	Y	RCD_TYPE
Process Status	To check the current status of the process. Default is P-Pending.		VARCHAR2	1	Y	PROCESS_STATUS
Message Category Number	Defined error message category		NUMBER	5,0	Y	MESSAGE_CAT_NBR

Table 3–4 (Cont.) Account Repayment Schedule

Field Name	Description	Value	Data Type	Length	Required	Column Name
Message Number	Error message number		NUMBER	5,0	Y	MESSAGE_NBR
Record Update Date	Date on which the record is updated		DATE	10	N	RECORD_UPDATE_DT
Record Exist Switch	To check whether record is already available or not		VARCHAR2	1	Y	RECORD_EXISTS_SW

3.1.1.5 Account Warning Indicator

Table Name: Account Warning Indicator (CI_FDR_ACCT_WARNING_IND)

Description: This table holds account warning indicators data from host.

Table 3–5 Account Warning Indicator

Field Name	Description	Value	Data Type	Length	Required	Column Name
Account No	Account Number as stored in Host		VARCHAR2	40	Y	HOST_ACCT_NBR
Host ID	Source Host ID for host		VARCHAR2	10	Y	SRC_HOST_ID
Warning Indicator Code	Warning Indicator code as stored in host		VARCHAR2	50	Y	WARN_IND_CD
Warning Indicator Value	Warning Indicator Value		VARCHAR2	1	N	WARN_IND_VAL
Start Date	Start Date for warning indicator		DATE	10	N	START_DT
End Date	End Date for the warning indicator code		DATE	10	N	END_DT
Record Creation Date	Date on which data is fed to Collections		DATE	10	N	CRET_DTTM
Record Type	Signifies if the data is created initially or is update for existing data	I - Insert U - Update	VARCHAR2	10	Y	RCD_TYPE
Process Status	To check the current status of process. Default is P-Pending.		VARCHAR2	1	Y	PROCESS_STATUS
Message Category	Defined error message category		NUMBER	5,0	Y	MESSAGE_CAT_NBR
Message Number	Error message number		NUMBER	5,0	Y	MESSAGE_NBR
Record Update Date	Date on which the record is updated		DATE	10	N	RECORD_UPDATE_DT
Record Exist Switch	To check whether record is already available or not		VARCHAR2	1	Y	RECORD_EXISTS_SW

3.1.2 Party Data

This section provides information on the tables related to party.

3.1.2.1 Party Account Relationship

Table Name: Party Account Relationship (CI_FDR_ACCT_PER)

Description: This table holds account party relationships data from host.

Table 3–6 Account Party Relationship

Field Name	Description	Value	Data Type	Length	Required	Column Name
Account Number	Account Number in Host		VARCHAR2	40	Y	HOST_ACCT_ NBR
Host ID	Source Host ID for host		VARCHAR2	10	Y	SRC_HOST_ID
Party ID	Party ID as stored in Host		VARCHAR2	40	Y	HOST_CUST_ NBR
Account Relationship	Account Relationship Code	Sole Owner, Joint and First, Joint and Others, Trustee Auth Signatory and Power of Attorney	CHAR	8	Y	ACCT_REL_ TYPE_CD
Phone Banking Flag	This flag signifies if the phone banking flag is enabled for the customer account relationship if maintained at this level.		VARCHAR2	1	N	FDR_PHONE_ BANK_SW
Internet Banking Flag	This flag signifies if the internet banking flag is enabled for the customer account relationship if maintained at this level.		VARCHAR2	1	N	FDR_ INTERNET_ BANK_SW
Mobile Banking Flag	This flag signifies if the mobile banking flag is enabled for the customer account relationship if maintained at this level.		VARCHAR2	1	N	FDR_MOBILE_ BANK_SW
ATM Card Flag	This flag signifies if the ATM Card has been issued to the customer for this account.		VARCHAR2	1	N	FDR_ATM_SW
Debit Card Flag	This flag signifies if the Debit Card has been issued to the customer for this account.		VARCHAR2	1	N	FDR_ DEBITCARD_ SW
UDF1	User Defined Fields		VARCHAR2	60	N	UDF1
UDF2	User Defined Fields		VARCHAR2	60	N	UDF2
UDF3	User Defined Fields		VARCHAR2	60	N	UDF3
UDF4	User Defined Fields		VARCHAR2	60	N	UDF4
UDF5	User Defined Fields		VARCHAR2	60	N	UDF5

Table 3–6 (Cont.) Account Party Relationship

Field Name	Description	Value	Data Type	Length	Required	Column Name
UDF6	User Defined Fields		VARCHAR2	60	N	UDF6
UDF7	User Defined Fields		VARCHAR2	60	N	UDF7
UDF8	User Defined Fields		VARCHAR2	60	N	UDF8
UDF9	User Defined Fields		VARCHAR2	60	N	UDF9
UDF10	User Defined Fields		VARCHAR2	60	N	UDF10
Record Creation Date	Date on which data is fed to Collections		DATE	10	N	CRET_DTTM
Record Type	Signifies if the data is created initially or is update for existing data	I - Insert U - Update	VARCHAR2	10	Y	RCD_TYPE
Account Nick Name	Account Nick Name		VARCHAR2	120	N	ACCT_NICKNAME
CORRES_NOMINATION_SW	Correspondence nomination switch		CHAR	1	N	CORRES_NOMINATION_SW
Process Status	To check the current status of the process. Default is P-Pending.		VARCHAR2	1	Y	PROCESS_STATUS
Message Category Number	Defined error message category		NUMBER	5,0	Y	MESSAGE_CAT_NBR
Message Number	Error message number		NUMBER	5,0	Y	MESSAGE_NBR
Record Update Date	Date on which the record is updated		DATE	10	N	RECORD_UPDATE_DT
Record Exist Switch	To check whether the record is available or not		VARCHAR2	1	Y	RECORD_EXISTS_SW
RMB main customer	RMB main customer		CHAR	1	N	RMB_MAIN_CUST
Financial Responsible switch	Financial Responsible switch		CHAR	1	N	RMB_FIN_RESP

3.1.2.2 Party Details

Table Name: Party Details (CI_FDR_PER)

Description: This table holds party data from host.

Table 3–7 Party Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Feeder Person Id			VARCHAR2	10	Y	FDR_PER_ID
Party ID	Party ID as stored in Host		VARCHAR2	40	Y	HOST_CUST_NBR
Host ID	Source Host ID for host		VARCHAR2	10	Y	SRC_HOST_ID

Table 3–7 (Cont.) Party Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Determinant Value	Determinant Value for identification of Party. This will depend on setups in host and is used in case of multi-branding features.		VARCHAR2	60	Y	DETERMINAN T_VALUE
Party Class	This field displays the party class of the customer. Party Class is a sub category in the Party Type. Fixed values for Individual party type are: Salaried Self Employed		VARCHAR2	40	N	PER_CL_CD
Date of Birth / Date of Incorporation/ Date of Trust Deed			DATE	10	N	BIRTH_DT
Marital Status	Marital Status of Party in case of Individual Customer		VARCHAR2	20	N	MARITAL_ STAT_FLG
Customer Since			DATE	10	N	SETUP_DT
Gender	Gender of Individual Customer		VARCHAR2	4	N	GENDER
Preferred Language	Preferred Language of Communication		VARCHAR2	3	N	LANGUAGE_ CD
Marketing Info Flag	Marketing Information Flag to continue communication		VARCHAR2	4	N	FDR_RECV_ MKTG_INFO_ FLG
Probability of Default	String value coming from third party interface		VARCHAR2	60	N	PROBABILITY _OF_DEFLT_ VAL
3rd Party Flag	Indicates if a third party is associated to the party	Y/N	VARCHAR2	1	N	FDR_THIRD_ PARTY_SW
Internet Banking Flag	This flag signifies if internet banking flag is enabled for the customer	Y/N	VARCHAR2	1	N	FDR_ INTERNET_ BANK_SW
Phone Banking Flag	This flag signifies if phone banking flag is enabled for the customer	Y/N	VARCHAR2	1	N	FDR_PHONE_ BANK_SW
VIP Flag	This flag signifies if this is a VIP customer	Y/N	VARCHAR2	1	N	FDR_VIP_ PARTY_SW
Behavior Score	Also available at Customer Level - Numeric value coming from third party interface		VARCHAR2	10	N	FDR_ BEHAVIOR_ SCORE

Table 3–7 (Cont.) Party Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Customer Risk Score (CRS)	Customer Risk Score (CRS)		VARCHAR2	10	N	FDR_ CUSTOMER_ RISK_SCORE
Party Type	This field displays the party type. Valid values: - Individual - Corporate - Trust		VARCHAR2	10	Y	FDR_PER_OR_ BUS_FLG
User Defined Value 1	User Defined Fields		VARCHAR2	60	N	UDF1
User Defined Value 2	User Defined Fields		VARCHAR2	60	N	UDF2
User Defined Value 3	User Defined Fields		VARCHAR2	60	N	UDF3
User Defined Value 4	User Defined Fields		VARCHAR2	60	N	UDF4
User Defined Value 5	User Defined Fields		VARCHAR2	60	N	UDF5
User Defined Value 6	User Defined Fields		VARCHAR2	60	N	UDF6
User Defined Value 7	User Defined Fields		VARCHAR2	60	N	UDF7
User Defined Value 8	User Defined Fields		VARCHAR2	60	N	UDF8
User Defined Value 9	User Defined Fields		VARCHAR2	60	N	UDF9
User Defined Value 10	User Defined Fields		VARCHAR2	60	N	UDF10
User Defined Value 11	User Defined Fields		VARCHAR2	60	N	UDF11
User Defined Value 12	User Defined Fields		VARCHAR2	60	N	UDF12
User Defined Value 13	User Defined Fields		VARCHAR2	60	N	UDF13
User Defined Value 14	User Defined Fields		VARCHAR2	60	N	UDF14
User Defined Value 15	User Defined Fields		VARCHAR2	60	N	UDF15
User Defined Value 16	User Defined Fields		VARCHAR2	60	N	UDF16
User Defined Value 17	User Defined Fields		VARCHAR2	60	N	UDF17
User Defined Value 18	User Defined Fields		VARCHAR2	60	N	UDF18
User Defined Value 19	User Defined Fields		VARCHAR2	60	N	UDF19
User Defined Value 20	User Defined Fields		VARCHAR2	60	N	UDF20

Table 3–7 (Cont.) Party Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Record Creation Date	Date on which data is fed to Collections		DATE	10	N	CRET_DTTM
Record Type	Signifies if the data is created initially or is update for existing data	I - Insert U - Update	VARCHAR2	10	Y	RCD_TYPE
FDR_ABILITY_TO_PAY_FLG	Ability to pay		VARCHAR2	4	N	FDR_ABILITY_TO_PAY_FLG
REALIZN_STAT	Realization Stat		VARCHAR2	10	N	REALIZN_STAT
Process Status	To check the current status of the process. Default is P-Pending.		VARCHAR2	1	Y	PROCESS_STATUS
Message Category Number	Defined error message category		NUMBER	5,0	Y	MESSAGE_CAT_NBR
Message Number	Error message number		NUMBER	5,0	Y	MESSAGE_NBR
Record Update Date	Date on which the record is updated		DATE	10	N	RECORD_UPDATE_DT
Record Exist Switch	To check whether the record is available or not		VARCHAR2	1	Y	RECORD_EXISTS_SW
Enterprise customer number	OCH Number		VARCHAR2	60	N	FDR_ENTERPRISE_CUST_NBR

3.1.2.3 Party Address Details

Table Name: Party Address Details (CI_FDR_PER_ADDR)

Description: This table holds party address data from host.

Table 3–8 Party Address Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Party ID	Party ID as stored in Host		VARCHAR2	40	Y	HOST_CUST_NBR
Host ID	Source Host ID for host		VARCHAR2	10	Y	SRC_HOST_ID
Address Type	Address Type Code maintained in Host	Home, Business, Postal, Seasonal	VARCHAR2	20	Y	ADDR_TYPE_CD
Sequence ID	Sequence ID maintained in Host for each address type in case multiple addresses are maintained for same address type		VARCHAR2	40	Y	FDR_ADDR_SEQ_ID
Address 1	Address Line 1		VARCHAR2	120	N	ADDRESS_LINE1

Table 3–8 (Cont.) Party Address Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Address 2	Address Line 2		VARCHAR2	120	N	ADDRESS_ LINE2
Address 3	Address Line 3		VARCHAR2	120	N	ADDRESS_ LINE3
Address 4	Address Line 4		VARCHAR2	120	N	ADDRESS_ LINE4
City	City Code		VARCHAR2	50	N	CITY_CD
Country	Country Code		VARCHAR2	30	N	COUNTRY_CD
Post/ Zip/ Pin Code	Zip Code		VARCHAR2	30	N	ZIP_CD
Determinant Value	Determinant Value for identification of Party. This will depend on setups in host and is used in case of multi-branding features.		VARCHAR2	60	Y	DETERMINAN T_VALUE
Status	Active or Inactive status		VARCHAR2	60	N	STATUS
Record Creation Date	Date on which data is fed to Collections		DATE	10	N	CRET_DTTM
Record Type	Signifies if the data is created initially or is update for existing data	I - Insert U - Update	VARCHAR2	10	N	RCD_TYPE
EFFECTIVE_ DT	Effective date		DATE	10	Y	EFFECTIVE_ DT
FDR_STATE_ CD	State code		VARCHAR2	60	N	FDR_STATE_ CD
Process Status	To check the current status of the process. Default is P-Pending.		VARCHAR2	1	N	PROCESS_ STATUS
Message Category Number	Defined error message category		NUMBER	5,0	N	MESSAGE_ CAT_NBR
Message Number	Error message number		NUMBER	5,0	N	MESSAGE_ NBR
Record Update Date	Date on which the record is updated		DATE	10	N	RECORD_ UPDATE_DT
Record Exist Switch	To check whether the record is available or not		VARCHAR2	1	Y	RECORD_ EXISTS_SW
SEASON_ START_MMDD	Season start month and day		VARCHAR2	4	N	SEASON_ START_MMDD
SEASON_ END_MMDD	Season end month and day		VARCHAR2	4	N	SEASON_ END_MMDD

3.1.2.4 Party Employment Details

Table Name: Party Employment Details (CI_FDR_PER_EMPLOYMENT_PROF)

Description: This table holds party employment details from host.

Table 3–9 Party Employment Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Party ID	Party ID as stored in Host		VARCHAR2	40	Y	HOST_CUST_ NBR
Host ID	Source Host ID for host		VARCHAR2	10	Y	SRC_HOST_ID
Determinant Value	Determinant Value for identification of Party. This will depend on setups in host and is used in case of multi-branding features.		VARCHAR2	60	Y	DETERMINAN T_VALUE
Sequence ID	Sequence ID of Employment details		VARCHAR2	40	Y	FDR_EMP_ SEQ_ID
Employment Status	Employment Status Code	**Employment Status:** For example:, Full Time, Part Time, Home Duties, Non-Resident, Pensioner, Retired, Student, Superannuati on, Unemployed	VARCHAR2	4	N	EMPLOYMEN T_STAT_CD
Employment Type	Employment Type	**Employment Type:** For example, Others, Salaried, Self Employed, Both- Salaried and Self Employed	VARCHAR2	30	N	EMPLOYMEN T_TYPE
Employer Name	Name of the employer of the customer		VARCHAR2	120	N	EMPLOYER_ NAME
Industry Type	Industry Type		VARCHAR2	30	N	INDUSTRY_ TYPE
Company Type		For example, Public Limited, Private Limited, Government Organization	VARCHAR2	30	N	COMPANY_ TYPE
Occupation	Occupation		VARCHAR2	30	N	PROFESSION_ TYPE
Designation	Designation		VARCHAR2	120	N	DESIGNATION _TXT

Table 3–9 (Cont.) Party Employment Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Gross Annual Salary	Gross Annual Salary		NUMBER	36,18	N	GRS_ ANNUAL_ INCOME
Start Date	Start Date		DATE	10	N	START_DT
End Date	End Date		DATE	10	N	END_DT
Status	Status		VARCHAR2	60	N	STATUS
Record Creation Date	Date on which data is fed to Collections		DATE	10	N	CRET_DTTM
Record Type	Signifies if the data is created initially or is update for existing data	I - Insert U - Update	VARCHAR2	10	Y	RCD_TYPE
Process Status	To check the current status of the process. Default is P-Pending.		VARCHAR2	1	Y	PROCESS_ STATUS
Message Category Number	Defined error message category		NUMBER	5,0	Y	MESSAGE_ CAT_NBR
Message Number	Error message number		NUMBER	5,0	Y	MESSAGE_ NBR
Record Update Date	Date on which the record is updated		DATE	10	N	RECORD_ UPDATE_DT
Record Exist Switch	To check whether the record is available or not		VARCHAR2	1	Y	RECORD_ EXISTS_SW

3.1.2.5 Party Identification Details

Table Name: Party Identification Details (CI_FDR_PER_ID)

Description: This table holds party ID type details from host.

Table 3–10 Party Identification Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Party ID	Party ID as stored in Host		VARCHAR2	40	Y	HOST_CUST_ NBR
Host ID	Source Host ID for host		VARCHAR2	10	Y	SRC_HOST_ID
Identification Type	Value of Identification Type Code	Passport No, Driving License No, and so on.	VARCHAR2	30	Y	FDR_ID_TYPE
ID Value	Identification Number corresponding to each of the identification types		VARCHAR2	40	N	FDR_ID_NBR

Table 3–10 (Cont.) Party Identification Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Determinant Value	Determinant Value for identification of Party. This will depend on setups in host and is used in case of multi-branding features.		VARCHAR2	60	Y	FDR_DETERMINANT_VALUE
Issue Date	Issue Date for Identification Number		DATE	10	N	FDR_ISSUE_DT
Expiry Date	Expiry Date for Identification Number		DATE	10	N	FDR_EXPIRY_DT
Record Creation Date	Date on which data is fed to Collections		DATE	10	N	CRET_DTTM
Record Type	Signifies if the data is created initially or is update for existing data	I - Insert U - Update	VARCHAR2	10	Y	RCD_TYPE
Process Status	Used to check current status of process. Default is P-Pending.		VARCHAR2	1	Y	PROCESS_STATUS
Message Category Number	Defined error message category		NUMBER	5,0	Y	MESSAGE_CAT_NBR
Message Number	Error message number		NUMBER	5,0	Y	MESSAGE_NBR
Record Update Date	Date on which the record is updated		DATE	10	N	RECORD_UPDATE_DT
Record Exist Switch	To check whether the record is available or not		VARCHAR2	1	Y	RECORD_EXISTS_SW
ID_TYPE_VAL_STATUS	ID Type Status		VARCHAR2	10	N	ID_TYPE_VAL_STATUS

3.1.2.6 Party Name Details

Table Name: Party Name Details (CI_FDR_PER_NAME)

Description: This table holds party name details from host.

Table 3–11 Party Name Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Party ID	Party ID as stored in Host		VARCHAR2	40	Y	HOST_CUST_NBR
Host ID	Source Host ID for host		VARCHAR2	10	Y	SRC_HOST_ID
Name Type	Type of Name	Legal	VARCHAR2	10	Y	FDR_NAME_TYPE
First Prefix	Indicates the first prefix		VARCHAR2	30	N	FDR_FIRST_PREFIX_ID
Second Prefix	Indicates the second prefix		VARCHAR2	30	N	FDR_SECOND_PREFIX_ID

Table 3–11 (Cont.) Party Name Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
First Name	First Name of the customer		VARCHAR2	50	N	FDR_FIRST_ NAME
First Middle Name	First middle name of the customer		VARCHAR2	50	N	FDR_MIDDLE_ NAME_FIRST
Second Middle Name	Second Middle name of the customer		VARCHAR2	50	N	FDR_MIDDLE_ NAME_ SECOND
Last Name	Last Name of the customer		VARCHAR2	50	N	FDR_LAST_ NAME
Suffix ID	Suffix ID in the name		VARCHAR2	30	N	FDR_SUFFIX_ ID
Full Name	Full name of the customer		VARCHAR2	250	N	FDR_FULL_ NAME
Short Name	Short Name of the customer		VARCHAR2	60	N	FDR_SHORT_ NAME
Determinant Value	Determinant Value for identification of Party. This will depend on setups in host and is used in case of multi-branding features.		VARCHAR2	60	Y	FDR_ DETERMINAN T_VALUE
Primary Name Flag	Signifies if a particular name needs to be used as a primary name for the customer	Y/N	CHAR	1	N	FDR_ PRIMARY_ NAME_SW
Record Creation Date	Date on which data is fed to Collections		DATE	10	N	CRET_DTTM
Record Type	Signifies if the data is created initially or is update for existing data	I - Insert U - Update	VARCHAR2	10	Y	RCD_TYPE
PER_NAME_ STATUS	Person name status		VARCHAR2	10	N	PER_NAME_ STATUS
Process Status	To check the current status of the process. Default is P-Pending.		VARCHAR2	1	Y	PROCESS_ STATUS
Message Category Number	Defined error message category		NUMBER	5,0	Y	MESSAGE_ CAT_NBR
Message Number	Error message number		NUMBER	5,0	Y	MESSAGE_ NBR
Record Update Date	Date on which the record is updated		DATE	10	N	RECORD_ UPDATE_DT

Table 3–11 (Cont.) Party Name Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Record Exist Switch	To check whether the record is available or not		VARCHAR2	1	Y	RECORD_ EXISTS_SW
FIRST_ PREFIX_DESC	First name prefix		VARCHAR2	120	N	FIRST_ PREFIX_DESC
SECOND_ PREFIX_DESC	Second name prefix		VARCHAR2	120	N	SECOND_ PREFIX_DESC

3.1.2.7 Party Contact Preference Details

Table Name: Party Contact Preference Details (CI_FDR_CONTACT_PREF)

Description: This table holds the party contact preference data from host.

Table 3–12 Party Contact Preference Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Party ID	Party ID as stored in Host		VARCHAR2	40	Y	HOST_CUST_ NBR
Host ID	Source Host ID for host		VARCHAR2	10	Y	SRC_HOST_ID
Determinant Value	Determinant Value for identification of Party. This will depend on setups in host and is used in case of multi-branding features.		VARCHAR2	60	Y	DETERMINAN T_VALUE
Contact Point	Type of Contact Point	Mobile, Landline, Email, and so on.	VARCHAR2	10	Y	CONTACT_ POINT_TYPE
Purpose			VARCHAR2	120	N	PURPOSE_TXT
Value	Contact Point Value, for example, if Contact Point is Mobile then provide mobile number, if Email then provide email ID		VARCHAR2	400	N	CONTACT_ VALUE
Contact Type		Home, Work, Others	VARCHAR2	10	Y	CONTACT_ PREF_TYPE
Start Date	Start date for using this contact point and type		DATE	10	N	START_DT
End Date	End date for using this contact point and type		DATE	10	N	END_DT
Time From (weekdays)	Start Time for contacting on weekdays	In hundred hour format (for example, 1800 for 6:00 PM)	NUMBER	10,0	N	WKDAY_ FROM_TM

Table 3–12 (Cont.) Party Contact Preference Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Time To (weekdays)	End Time for contacting on weekdays	In hundred hour format (for example, 1800 for 6:00 PM)	NUMBER	10,0	N	WKDAY_TO_ TM
Time From (weekends)	Start Time for contacting on weekends	In hundred hour format (for example, 1800 for 6:00 PM)	NUMBER	10,0	N	WKEND_ FROM_TM
Time To (weekends)	End Time for contacting on weekends	In hundred hour format (for example, 1800 for 6:00 PM)	NUMBER	10,0	N	WKEND_TO_ TM
Preference Frequency	Preferred Frequency of contact		NUMBER	20	N	PREFERENCE_ FREQUENCY
Primary Contact Point	Primary Contact Point Flag		VARCHAR2	10	N	FDR_ PRIMARY_SW
Status	Status - if Active or Dormant		VARCHAR2	60	Y	STATUS
Record Creation Date	Date on which data is fed to Collections		DATE	10	N	CRET_DTTM
Record Type	Signifies if the data is created initially or is update for existing data	I - Insert U - Update	VARCHAR2	10	N	RCD_TYPE
Process Status	To check the current status of the process. Default is P-Pending.		VARCHAR2	1	Y	PROCESS_ STATUS
Message Category Number	Defined error message category		NUMBER	5,0	Y	MESSAGE_ CAT_NBR
Message Number	Error message number		NUMBER	5,0	Y	MESSAGE_ NBR
Record Update Date	Date on which the record is updated		DATE	10	N	RECORD_ UPDATE_DT
Record Exist Switch	To check whether the record is available or not		VARCHAR2	1	Y	RECORD_ EXISTS_SW

3.1.2.8 Party Warning Indicators

Table Name: Party Warning Indicators (CI_FDR_PARTY_WARNING_IND)

Description: This table holds the party warning indicators data from host.

Table 3–13 Party Warning Indicators

Field Name	Description	Value	Data Type	Length	Required	Column Name
Party ID	Party ID as stored in Host		VARCHAR2	40	Y	HOST_CUST_ NBR
Host ID	Source Host ID for host		VARCHAR2	10	Y	SRC_HOST_ID
Warning Indicator Code	Warning Indicator Code		VARCHAR2	50	Y	WARN_IND_ CD
Warning Indicator Value	Value of Warning Indicator Code	Y/N	VARCHAR2	1	N	WARN_IND_ VAL
Start Date	Start Date of Warning Indicator		DATE	10	N	START_DT
End Date	End Date of warning Indicator		DATE	10	N	END_DT
Record Creation Date	Date on which data is fed to Collections		DATE	10	N	CRET_DTTM
Record Type	Signifies if the data is created initially or is update for existing data	I - Insert U - Update	VARCHAR2	10	Y	RCD_TYPE
Process Status	To check the current status of the process. Default is P-Pending.		VARCHAR2	1	Y	PROCESS_ STATUS
Message Category Number	Defined error message category		NUMBER	5,0	Y	MESSAGE_ CAT_NBR
Message Number	Error message number		NUMBER	5,0	Y	MESSAGE_ NBR
Record Update Date	Date on which the record is updated		DATE	10	N	RECORD_ UPDATE_DT
Record Exist Switch	To check whether the record is available or not		VARCHAR2	1	Y	RECORD_ EXISTS_SW

3.1.3 Collateral Data

This section provides information on the tables related to collaterals.

3.1.3.1 Collateral Details

Table Name: Collateral Details (CI_FDR_COLLATERAL)

Description: This table holds collateral data from host.

Table 3–14 Collateral Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Collateral Code	Collateral Code as stored in host		VARCHAR2	40	Y	COLLATERAL _CD
Host ID	Source Host ID for host		VARCHAR2	10	Y	SRC_HOST_ID
Collateral Type	Type of Collateral		VARCHAR2	50	N	COLLATERAL _TYPE

Table 3–14 (Cont.) Collateral Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Collateral Sub Type	If there are any collateral sub type		VARCHAR2	50	N	COLLATERAL _SUB_TYPE
Collateral Category	Collateral Category		VARCHAR2	50	N	COLLATERAL _CAT
Collateral Description	Collateral Description		VARCHAR2	300	N	FDR_ COLLATERAL _DESCR
Nature	Normal/ Guarantee		VARCHAR2	40	N	COLLATERAL _NATURE
Collateral Currency	Collateral Currency		VARCHAR2	3	N	COLLATERAL _CUR
Assessed Value	Market Value		NUMBER	36,18	N	ASSESD_ VALUE
Assessment Date	Date of assessment		DATE	10	N	ASSESD_DT
Bank Value	Book Value		NUMBER	36,18	N	BANK_VALUE
Sold By	This property is required to identify entity which sold the collateral.	Customer (Borrower), Bank, Court	VARCHAR2	255	N	SOLD_BY
Date of Sale	Date on which the collateral was sold		DATE	10	N	SALE_DT
Amount Realized	Gross Sale amount		NUMBER	36,18	N	AMT_ REALIZED
Date of Settlement	Date on which settlement took place		DATE	10	N	SETLMNT_DT
Realization Status	Final status of realization		VARCHAR2	60	N	REALIZATION _STATUS
Amount Recovered	Gross Sale Amount less Costs incurred for sale of collateral		NUMBER	36,18	N	FDR_AMT_ RECOVERED
Collateral Address Line1	Collateral Address Line1		VARCHAR2	120	N	ADDRESS_ LINE1
Collateral Address Line2	Collateral Address Line2		VARCHAR2	120	N	ADDRESS_ LINE2
Collateral Address Line3	Collateral Address Line3		VARCHAR2	120	N	ADDRESS_ LINE3
Collateral Address Line4	Collateral Address Line4		VARCHAR2	120	N	ADDRESS_ LINE4
City code	City code		VARCHAR2	50	N	CITY_CD
Postal code	Postal code		VARCHAR2	30	N	ZIP_CD
State code	State code		VARCHAR2	6	N	STATE_CD
Country code	Country code		VARCHAR2	30	N	COUNTRY_CD
Record Creation Date	Date on which data is fed to Collections		DATE	10	N	CRET_DTTM

Table 3–14 (Cont.) Collateral Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Record Type	Signifies if the data is created initially or is update for existing data	I - Insert U - Update	VARCHAR2	10	Y	RCD_TYPE
Process Status	To check the current status of the process. Default is P-Pending.		VARCHAR2	1	Y	PROCESS_ STATUS
Message Category Number	Defined error message category		NUMBER	5,0	Y	MESSAGE_ CAT_NBR
Message Number	Error message number		NUMBER	5,0	Y	MESSAGE_ NBR
Record Update Date	Date on which the record is updated		DATE	10	N	RECORD_ UPDATE_DT
Record Exist Switch	To check whether the record is available or not		VARCHAR2	1	Y	RECORD_ EXISTS_SW
Realization ID	Realization ID		VARCHAR2	50	N	REALIZATION _ID

3.1.3.2 Collateral Charge Details

Table Name: Collateral Charge Details (CI_FDR_COLLATERAL_CHRG)

Description: This table holds collateral charges details from host.

Table 3–15 Collateral Charges Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Collateral Code	Collateral Code as stored in host		VARCHAR2	40	Y	COLLATERAL _CD
Host ID	Source Host ID for host		VARCHAR2	10	Y	SRC_HOST_ID
Charge Code	Charge Codes maintained in the host		VARCHAR2	20	Y	CHRG_CD
Bank Value Relied On	Bank value for each of the charge codes		NUMBER	36,18	Y	AVL_ CHARGE_VAL
Charge Currency	Currency in which Charge Value is calculated. Collateral currency and charge currency can differ		CHAR	3	Y	CHARGE_ CURR
Record Creation Date	Date on which data is fed to Collections		DATE	10	N	CRET_DTTM
Record Type	Signifies if the data is created initially or is update for existing data	I - Insert U - Update	VARCHAR2	10	Y	RCD_TYPE
Process Status	To check the current status of the process. Default is P-Pending.		VARCHAR2	1	Y	PROCESS_ STATUS

Table 3–15 (Cont.) Collateral Charges Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Message Category Number	Defined error message category		NUMBER	5,0	Y	MESSAGE_ CAT_NBR
Message Number	Error message number		NUMBER	5,0	Y	MESSAGE_ NBR
Record Update Date	Date on which the record is updated		DATE	10	N	RECORD_ UPDATE_DT
Record Exist Switch	To check whether the record is available or not		VARCHAR2	1	Y	RECORD_ EXISTS_SW

3.1.3.3 Collateral Entity Mapping

Table Name: Collateral Entity Mapping (CI_FDR_COLLATERAL_ENTITY)

Description: This table holds the collateral entity mapping from host. Collateral can be mapped to facility or to an account.

Table 3–16 Collateral Entity Mapping

Field Name	Description	Value	Data Type	Length	Required	Column Name
Collateral Code	Collateral Code as stored in host		VARCHAR2	40	Y	COLLATERAL _CD
Host ID	Source Host ID for host		VARCHAR2	10	Y	SRC_HOST_ID
Entity Type	Entity to which collateral is mapped	ACCOUNT, FACILITY	VARCHAR2	10	Y	ENTITY_TYPE
Entity ID	Entity ID of entity to which collateral is mapped		VARCHAR2	40	Y	COL_ENTITY_ ID
Contribution Switch	Identify if the collateral is contributing towards an entity	Y/N	VARCHAR2	1	N	FDR_LIMIT_ CONTRIBUTIO N_SW
Record Creation Date	Date on which data is fed to Collections		DATE	10	N	CRET_DTTM
Record Type	Signifies if the data is created initially or is update for existing data	I - Insert U - Update	VARCHAR2	10	Y	RCD_TYPE
Process Status	To check the current status of the process. Default is P-Pending.		VARCHAR2	1	Y	PROCESS_ STATUS
Message Category Number	Defined error message category		NUMBER	5,0	Y	MESSAGE_ CAT_NBR
Message Number	Error message number		NUMBER	5,0	Y	MESSAGE_ NBR

Table 3–16 (Cont.) Collateral Entity Mapping

Field Name	Description	Value	Data Type	Length	Required	Column Name
Record Update Date	Date on which the record is updated		DATE	10	N	RECORD_ UPDATE_DT
Record Exist Switch	To check whether the record is available or not		VARCHAR2	1	Y	RECORD_ EXISTS_SW
Charge Code	Charge Codes maintained in the host		VARCHAR2	20	N	CHRG_CD

3.1.3.4 Collateral Guarantor Mapping

Table Name: Collateral Guarantor Mapping (CI_FDR_COLLATERAL_GRNTR)

Description: This table holds the guarantors data for the collateral.

Table 3–17 Collateral Guarantor Mapping

Field Name	Description	Value	Data Type	Length	Required	Column Name
Collateral Code	Collateral Code as stored in host		VARCHAR2	40	Y	COLLATERAL _CD
Host ID	Source Host ID for host		VARCHAR2	10	Y	SRC_HOST_ID
Party ID	Party ID of the guarantor		VARCHAR2	40	Y	HOST_CUST_ NBR
Record Creation Date	Date on which data is fed to Collections		DATE	10	N	CRET_DTTM
Record Type	Signifies if the data is created initially or is update for existing data	I - Insert U - Update	VARCHAR2	10	Y	RCD_TYPE
Process Status	To check the current status of the process. Default is P-Pending.		VARCHAR2	1	Y	PROCESS_ STATUS
Message Category Number	Defined error message category		NUMBER	5,0	Y	MESSAGE_ CAT_NBR
Message Number	Error message number		NUMBER	5,0	Y	MESSAGE_ NBR
Record Update Date	Date on which the record is updated		DATE	10	N	RECORD_ UPDATE_DT
Record Exist Switch	To check whether the record is available or not		VARCHAR2	1	Y	RECORD_ EXISTS_SW

3.1.3.5 Collateral Owner Mapping

Table Name: Collateral Owner Mapping (CI_FDR_COLLATERAL_PARTY)

Description: This table holds ownership of parties for the collateral.

Table 3–18 Collateral Owner Mapping

Field Name	Description	Value	Data Type	Length	Required	Column Name
Collateral Code	Collateral Code as stored in host		VARCHAR2	40	Y	COLLATERAL _CD
Host ID	Source Host ID for host		VARCHAR2	10	Y	SRC_HOST_ID
Party ID	Party ID of Customer mapped to collateral		VARCHAR2	40	Y	HOST_CUST_ NBR
Percentage of Ownership	Ownership Percentage of each of the Party		VARCHAR2	10	N	OWNERSHIP_ PERCENT
Record Creation Date	Date on which data is fed to Collections		DATE	10	N	CRET_DTTM
Record Type	Signifies if the data is created initially or is an update for existing data	I - Insert U - Update	VARCHAR2	10	Y	RCD_TYPE
Process Status	To check the current status of the process. Default is P-Pending.		VARCHAR2	1	Y	PROCESS_ STATUS
Message Category Number	Defined error message category		NUMBER	5,0	Y	MESSAGE_ CAT_NBR
Message Number	Error message number		NUMBER	5,0	Y	MESSAGE_ NBR
Record Update Date	Date on which the record is updated		DATE	10	N	RECORD_ UPDATE_DT
Record Exist Switch	To check whether the record is available or not		VARCHAR2	1	Y	RECORD_ EXISTS_SW

3.1.4 Insurance Data

This section provides information on the tables related to insurance.

3.1.4.1 Insurance Details

Table Name: Insurance Details (CI_FDR_INSR_DTLS)

Description: This table holds insurance records for collateral, party, or facility.

Table 3–19 Insurance Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Entity ID	Value of Entity ID		VARCHAR2	40	Y	COL_ENTITY_ ID
Entity Type	Entity on which Insurance is captured. Possible Values	COLLATER AL, PERSON, or FACILITY	VARCHAR2	10	Y	ENTITY_TYPE
Host ID	Source Host ID for host		VARCHAR2	10	Y	SRC_HOST_ID
Insurance ID	Insurance ID as stored in host		VARCHAR2	60	Y	INSURANCE_ ID
Policy No	Policy number of the Insurance		VARCHAR2	50	Y	POLICY_NUM

Table 3–19 (Cont.) Insurance Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Insurance Policy Name	Insurance Policy Name		VARCHAR2	100	N	FDR_ INSURANCE_ POLICY_ NAME
Insured Currency	Currency Code of the Insured Amount		VARCHAR2	3	N	INSURED_ CURR
Insured Amount	Insured Amount		NUMBER	36,18	N	INSURED_ AMT
Insurer Code	Insurer Code as stored in host		VARCHAR2	50	N	INSURER_CD
Insurer Name	Insurer Name as stored in host		VARCHAR2	64	N	INSURER_ NAME
Policy Start Date	Start date of Policy		DATE	10	N	POLICY_ START_DT
Policy End Date	End date of Policy		DATE	10	N	POLICY_END_ DT
Premium Amount	Insurance Premium		NUMBER	36,18	N	PREMIUM_ AMT
Payment Frequency	Premium payment frequency		VARCHAR2	30	N	PAYMENT_ FREQ
Insurance Type	Insurance Type	LMI PPI	VARCHAR2	30	N	INSURANCE_ TYPE
Record Creation Date	Date on which data is fed to Collections		DATE	10	N	CRET_DTTM
Record Type	Signifies if the data is created initially or is update for existing data	I - Insert U - Update	VARCHAR2	10	Y	RCD_TYPE
DUA_ APPLICABLE	A DUA Switch applicable for LMI Insurance		VARCHAR2	1	N	DUA_ APPLICABLE
NET_BORR_ PREMIUM_ AMOUNT	Net borrower premium amount		NUMBER	36,18	N	NET_BORR_ PREMIUM_ AMOUNT
FDR_PARTY_ ID	Party ID		VARCHAR2	40	Y	FDR_PARTY_ ID
Process Status	To check the current status of the process. Default is P-Pending.		VARCHAR2	1	Y	PROCESS_ STATUS
Message Category Number	Defined error message category		NUMBER	5,0	Y	MESSAGE_ CAT_NBR
Message Number	Error message number		NUMBER	5,0	Y	MESSAGE_ NBR
Record Update Date	Date on which the record is updated		DATE	10	N	RECORD_ UPDATE_DT
Record Exist Switch	To check whether the record is available or not		VARCHAR2	1	Y	RECORD_ EXISTS_SW

3.1.5 Payment Data

This section provides information on the tables related to payments.

3.1.5.1 Online Payment Records

Table Name: Online Payment (CI_FDR_PAYMENTS)

Description: This table holds the failed online payment records which is used by payment processing batch for offline processing.

Table 3–20 Online Payment

Field Name	Description	Value	Data Type	Length	Required	Column Name
Account No	Account Number as stored in Host		VARCHAR2	40	Y	HOST_ACCT_NBR
Host ID	Source Host ID for host		VARCHAR2	10	Y	SRC_HOST_ID
Transaction Reference Number	Transaction Reference Number for payment transaction in host		VARCHAR2	30	Y	XREF_NO
Transaction Date	Date of Transaction		DATE	10	N	FDR_TRANSACTION_DT
Transaction Time	Time for Transaction		DATE	10	N	FDR_TRANSACTION_TM
Value Date	Value Date on which the transaction was posted in the host		DATE	10	N	FDR_VALUE_DT
Transaction Currency	Currency code of the transaction		VARCHAR2	3	N	FDR_TRANSACTION_CURR_CD
Transaction Amount	Payment Amount		NUMBER	36,18	N	FDR_TRANSACTION_AMT
Account Currency	Account Currency Code		VARCHAR2	3	N	FDR_ACCT_CURR_CD
Account Balance	Account Balance after Payment		NUMBER	36,18	N	FDR_ACCT_AMT
Transaction Code	Transaction Code as captured in the host		VARCHAR2	30	N	FDR_TRANSACTION_CD
Narration Text	Narration text for the transaction		VARCHAR2	120	N	FDR_NARRATION_TXT
Transaction Type Flag	Identify if the transaction is Credit or Debit that is, actual payment transaction or reversal	C/D	CHAR	1	Y	FDR_TRANSACTION_TYPE_FLG
Record Creation Date	Date on which data is fed to Collections		DATE	10	N	CRET_DTTM
Original Transaction ref number	Used for cancellation of payments		VARCHAR2	30	N	ORIG_XREF_NO

Table 3–20 (Cont.) Online Payment

Field Name	Description	Value	Data Type	Length	Required	Column Name
Transaction sequence number	Transaction sequence number		VARCHAR2	30	Y	FDR_XREF_SUB_SEQ_NO
Original Transaction sequence number	Used for cancellation of payments		VARCHAR2	30	N	FDR_ORIG_XREF_SUB_SEQ_NO
Process Status	To check the current status of the process. Default is P-Pending.		VARCHAR2	1	Y	PROCESS_STATUS
Message Category Number	Defined error message category		NUMBER	5,0	Y	MESSAGE_CAT_NBR
Message Number	Error message number		NUMBER	5,0	Y	MESSAGE_NBR

3.1.6 Group Data

This section provides information on the tables related to groups.

3.1.6.1 Group Details

Table Name: Group Details

Description: This table holds group related data from host.

Table 3–21 Group Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Group Id	Group id stored in Host		VARCHAR2	15	Y	GRP_ID
Determinant Value	Determinant Value for the Group		VARCHAR2	60	Y	DETERMINANT_VALUE
Host ID	Source Host ID for host		VARCHAR2	10	Y	SRC_HOST_ID
Group Name	Group name as defined in host. This field holds the name of the group.		VARCHAR2	250	N	GRP_NAME
Group Short Name	Group short name as defined in host. This field holds the short name of the group.		VARCHAR2	50	N	GRP_SHORT_NAME
Group RM	Group RM holds group Relationship Manager name defined in host		VARCHAR2	40	N	GRP_RM_NAME
Group Type	Group Type defined in host		VARCHAR2	40		GRP_TYPE
Watch List Flag	Watch list flag defined in host. Indicates that whether group ready to fall in collection for further processing		CHAR	1	Y	WATCHLIST_SW

Table 3–21 (Cont.) Group Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Record Creation Date	Date on which data is fed to Collections		DATE		Y	CRET_DTTM
Record Update Date	Date on which Record is updated		DATE		N	RECORD_ UPDATE_DT
Record Type	Signifies if the Member is added initially or is update for existing Member	I - Insert U - Update	VARCHAR2	10	Y	RCD_TYPE
Process Status	Used to check current status of process, Default is "P"- Pending		VARCHAR2	1	Y	PROCESS_ STATUS
Group Object Status	If group is reopened then required value will be sent in this column.		CHAR	1	Y	GRP_OBJ_ STATUS

3.1.6.2 Group Member Details

Table Name: Group Member Detail (CI_FDR_GRP_MEMBER)

Description: This table holds group member related data from host.

Table 3–22 Group Member Details

Field Name	Description	Value	Data Type	Length	Required	Column Name
Group Id	Group id stored in Host		VARCHAR2	15	Y	GRP_ID
Determinant Value	Determinant Value for the Group		VARCHAR2	60	Y	DETERMINAN T_VALUE
Host ID	Source Host ID for host		VARCHAR2	10	Y	SRC_HOST_ID
Party ID	Member id (Party Id/Host customer number) of the group		VARCHAR2	40	Y	HOST_CUST_ NBR
Party Name	Member/party Name		VARCHAR2	250	N	PARTY_NAME
Record Creation Date	Date on which data is fed to Collections		DATE		N	CRET_DTTM
Record Type	Signifies if the Member is added initially or is update for existing Member or deleting member	I - Insert U - Update D - Delete	VARCHAR2	10	Y	RCD_TYPE
Process Status	Used to check current status of process, Default is "P"- Pending		VARCHAR2	1	Y	PROCESS_ STATUS
Record Update Date	Date on which Record is updated		DATE		N	RECORD_ UPDATE_DT

3.2 OBP Views

Collections system pulls delinquent account data from the following views provided by OBP.

3.2.1 Main Account Views

The main account views are as follows:

- FLX_COL_ACCT_DATA_XF
- FLX_LN_COL_FD_ACCT_VW
- FLX_DD_COL_DATA_TOD_XF_VW
- FLX_DD_COL_DATA_XF_VW
- FLX_AC_COL_FD_ACCT_ARS_VW
- FLX_LN_COL_FD_SCH_VW
- FLX_COL_ACCT_WARN_IND_DATA_XF

3.2.2 Account Updateable Views

The account updateable views are as follows:

- FLX_DD_COL_DATA_XF_UPD_VW
- FLX_LN_COL_ACCT_UPDATE_VW

3.2.3 Hardship Views

The hardship views are as follows:

- FLX_COL_ACCT_HRDSHIP_VW
- FLX_LN_COL_ACCT_HRDSHIP_VW
- FLX_DD_COL_ACCT_HRDSHIP_VW

3.2.4 Party Views

The party views are as follows:

- FLX_PI_COL_FD_ACCT_PER_VW
- FLX_PI_COL_FD_PER_VW
- FLX_PI_COL_FD_PARTY_IDENT_VW
- FLX_PI_COL_FD_PER_NAME_VW
- FLX_PI_COL_FD_PER_WARN_IND_VW
- FLX_PI_COL_FD_EMP_PROF_VW
- FLX_PI_COL_FD_PER_ADDR_VW
- FLX_PI_COL_FD_CONTACT_PREF_VW

3.2.5 LCM / Collateral Views

The LCM / Collateral views are as follows:

- FLX_LM_COL_FD_COL_ENTITY_VW
- FLX_LM_COL_FD_COLLATERAL_VW
- FLX_LM_COL_FD_COL_PARTY_VW
- FLX_LM_COL_FD_COL_CHRG_VW

- FLX_LM_COL_FD_COL_GRNTR_VW
- FLX_LM_COL_FD_INSR_DTLS_VW

3.2.6 Group Views

The group views are as follows:

- FLX_PI_COL_FD_GROUP_V
- FLX_PI_COL_FD_GRP_MEMBER_V

4 Algorithms

This chapter provides information about list of algorithm types shipped out for OBP Collections.

4.1 Stop Contract: C1-CURENTITY

This section provides details of the Stop Contract: C1-CURENTITY algorithm.

Table 4–1 Stop Contract: C1-CURENTITY

Description	This algorithm type is used to stop the contract.
Detailed Description	Contract Stop Algorithm
Algorithm Entity	Cure Entity
Program Type	Java
Program Name	com.splwg.ccb.domain.collection.batch.algorithm.CureEntityAlgorithm
Parameters	NA
	This algorithm invokes the C1-StopServiceAgreement business service to set contract status as STOPPED. The contract end date is specified as system date.

4.2 Cure Account: C1-FINCOLL

This section provides details of the Cure Account: C1-FINCOLL algorithm.

Table 4–2 Cure Account: C1-FINCOLL

Description	This algorithm is used to invoke the OBP Services when contract is stopped during the finalize collection process.
Detailed Description	This algorithm type is used to invoke the OBP Services to update the delinquent flag=N when the contract is stopped during the finalize collection process.
Algorithm Entity	Contract Type - Contract Stop
Program Type	Java
Program Name	com.splwg.ccb.domain.collection.batch.algorithm.FinalizeCollectionContractStopAlgoComp

Table 4–2 (Cont.) Cure Account: C1-FINCOLL

Parameters	**Name:** contactMethods **Required (Yes/No):** Yes **Description:** Contact Methods soft parameter has a comma-separated value of customer contact methods. For example, SMS, EM, and so on. This value is used to calculate the number of self cured statistic.
Detailed Design	This algorithm invokes the OBP Services to update the delinquent flag =N and In collection flag = N in host (updateInCollectionIndicator()) when the contract is stopped during the final collection process. It also deletes the account review date from CI_ADM_RVW_SCH table, and updates the number of times an account is self-cured.

Table 4–3 Cure Account: Sample Algorithm

Algorithm Name	C1-FINCOL
Parameters	**Name:** contactMethods **Value:** SMS, EM

4.3 Queue Allocation: C1-ALLOCQUEU

This section provides details of the Queue Allocation: C1-ALLOCQUEU algorithm.

Table 4–4 Queue Allocation: C1-ALLOCQUEU

Description	Allocation algorithm for allocation cases to queue in round-robin method.
Detailed Description	This is an allocation algorithm for the allocation group to allocate cases to queues in round-robin method. This algorithm is invoked by the Allocation monitor batch (C1-ALOCM).
Algorithm Entity	Allocation Group -Queue Allocation
Program Type	Java
Program Name	Com.splwg.ccb.domain.collection.batch.algorithm.AllocationGroupQueueAlgoComp
Parameters	**Name:** queueAllocationView (soft parameter **Required (Yes/No):** Yes **Description:** View for allocation **Name:** qallocationGroup (hard parameter) **Required (Yes/No):** Yes **Description:** Allocation Group code
Detailed Design	This algorithm receives input as Allocation Group code from the batch. The view used to filter cases is accepted as an algorithm soft parameter. Product will ship CI_ALLOCATION_MONITOR_VW view. For the given allocation group code, it allocates cases to linked queues of the allocation group in round-robin method. For detailed process, see batch process (C1-ALOCM).

Table 4–5 Queue Allocation: Sample Algorithm

Algorithm Name	C1-ALLOCQUEU
Parameters	**Name:** queueAllocationView **Value:** CI_ALLOCATION_MONITOR_VW

4.4 Update Customer Switch: C1-CUSTSW

This section provides details of the Update Customer Switch: C1-CUSTSW algorithm.

Table 4–6 Update Customer Switch: C1-CUSTSW

Description	This algorithm is used to update the customer level case switch.
Detailed Description	This algorithm is used to update customer level case status on case enter processing. Customer Level Switch Name: Specify the customer level case status switch that should be updated. For example, BANKRUPT_SW, HARDSHIP_SW, IMPRISONED_SW, DECEASED_SW, ABSCONDING_SW, and so on.
Algorithm Entity	Case Type - Enter Status
Program Type	java
Program Name	com.splwg.ccb.domain.collection.batch.algorithm.CustomerLevelSwitchUpdateAlgorithm
Parameters	**Name:** Customer Level Switch Name **Required (Yes/No):** Yes **Description:** Name of column or switch to be processed **Name:** Switch Value **Required (Yes/No):** Yes **Description:** Y or N
Detailed Design	This algorithm updates the customer level switch. This algorithm is attached to the Case Type Enter Status algorithm spot. This soft parameter identifies the field that must be updated with a value. The Customer Level switch name soft parameter accepts the column name that must be updated with switch values as Y or N. You must create different algorithm for each field with the value and attach it to the case type enter status algorithm spot.

Table 4–7 Update Customer Switch: Sample Algorithm

Algorithm Name	C1-BRUPTSW
Parameters	**Name:** Customer Level Switch Name **Value:** BANKRUPT_SW **Name:** Switch Value **Value:** Y

4.5 Update Legal/Repo Switch: C1-LEREPOCT

This section provides details of the Update Legal/Repo Switch: C1-LEREPOCT algorithm.

Table 4–8 Update Legal/Repo Switch: C1-LEREPOCT

Description	This algorithm is used to update Legal and Repo case status on enter processing.
Detailed Description	Legal Repo Switch Name: Specify the Legal or Repo case switch column name of account extension For example, LEGAL_CASE_EXISTS_SW or REPO_CASE_EXISTS_SW, and so on. Switch Value: Please enter the switch value as Y or N.
Algorithm Entity	Case Type - Enter Status
Program Type	java
Program Name	com.splwg.ccb.domain.collection.batch.algorithm.RepoAndLegalCaseUpdateAlgorithm
Parameters	**Name:** Legal Repo Switch Name **Required (Yes/No):** Yes **Description:** Name of column or switch to be processed **Name:** Switch Value **Required (Yes/No):** Yes **Description:** Y or N
Detailed Design	This algorithm is created to update the Legal Case Switch and Repo Case Switch derived fields. This algorithm is attached to the Case Type Enter Status algorithm spot. The soft parameter is used to identify the fields that should be updated. For example, ■ If the case is Legal then pass Legal Repo Switch name as LEGAL_CASE_EXISTS_SW and switch value as Y and then attach this algorithm to case life cycle where you want to update the switch. ■ If the case is Repo then pass Legal Repo Switch name as REPO_CASE_EXISTS_SW and switch value as Y and then attach this algorithm to the case life cycle where you want to update the switch.

Table 4–9 Update Legal/Repo Switch: Sample Algorithm

Algorithm Name	C1-LEGALSW
Parameters	**Name:** Legal Repo Switch Name **Value:** LEGAL_CASE_EXISTS_SW **Name:** Switch Value **Value:** Y

4.6 User Allocation - Round Robin: C1-USRALCRR

This section provides details of the User Allocation - Round Robin: C1-USRALCRR algorithm.

Table 4–10 User Allocation - Round Robin: C1-USRALCRR

Description	This algorithm is used to allocate cases to users or teams in round-robin method.
Detailed Description	This algorithm is used to allocate cases to user or teams in round-robin method. This algorithm is invoked by the User Allocation batch (C1-USALC).
Algorithm Entity	User Allocation
Program Type	java
Program Name	com.splwg.ccb.domain.collection.batch.algorithm.UserAllocationRoundRobinAlgorithm
Parameters	NA
Detailed Design	This algorithm receives input as queue code. The computation logic is explained below: ■ A1 = Total allocation for the user or team across all queues. ■ B1 = Total capacity of the user or team. This has to be defined in user or collection team configuration. ■ C1 = B1 - A1 = Total available capacity of the user or team. ■ A2 = Existing allocation to the user or team for the current queue. ■ B2 = Capacity of the user or team for the queue. This is defined in queue master. ■ C2 = B2 - A2 = Total available capacity of the user or team for the current queue. ■ Available capacity of the user or team for the queue is lower of C1 and C2. ■ Get all cases which are allocated to the queue and: - Have no users or teams attached to it OR - Current allocated user or team does not have active association with the queue ■ Get available capacity for each user or team. ■ Allocate cases to users or teams in a round-robin manner starting with user with highest available capacity and then in decreasing order of capacity. ■ A count of freshly allocated cases should be maintained for each user or team. ■ Allocation to a particular user will be skipped if the user is on leave. ■ Allocation to a particular user or team will be skipped if count of newly allocated cases = available capacity. ■ If capacity of all users and teams are exhausted and there are still cases pending allocation, these should be allocated to exception user. There will be no check for exception user's/team's capacity. Exception user's expiry date will be checked against SC_USR_GRP_USR table.

4.7 User Allocation - % Based: C1-USRALCPR

This section provides details of the User Allocation - % Based: C1-USRALCPR algorithm.

Table 4–11 User Allocation - % Based: C1-USRALCPR

Description	This algorithm is used for allocating cases to users or teams in percentage-based method.
Detailed Description	This algorithm allocates cases to user or teams in percentage-based method. This algorithm is invoked from the User Allocation batch (C1-USALC).
Algorithm Entity	User Allocation

Table 4–11 (Cont.) User Allocation - % Based: C1-USRALCPR

Program Type	java
Program Name	com.splwg.ccb.domain.collection.batch.algorithm.UserAllocationPercentageBaseAlgorithm
Parameters	NA
Detailed Design	This algorithm takes input as Queue code. The computation logic is as below: ▪ A1 = Total allocation for the user or team across all queues. ▪ B1 = Total capacity of the user or team. This has to be defined in user or collection team configuration. ▪ C1 = B1 - A1 = Total available capacity of the user or team. ▪ Available capacity of the user or team for the queue is C1. ▪ Get all cases which are allocated to the queue and - Have no users or teams attached to it OR - Current allocated user or team does not have active association with the queue ▪ Calculate % allocation for each user or team in the queue to find maximum cases of new cases that can be allocated to each user or team. ▪ Get "available capacity" for each user or team ▪ Allocate cases to users or teams in sequential manner starting with user with highest available capacity and then in decreasing order of capacity. ▪ A count of freshly allocated cases should be maintained for each user or team ▪ Allocation to a particular user will be skipped if the user is on leave. ▪ Allocation to a particular user or team will be skipped if count of newly allocated cases = available capacity. ▪ If capacity of all users and teams are exhausted and there are still cases pending allocation, these should be allocated to exception user or team. There will be no check for exception user's capacity. Exception user's expiry date will be checked against SC_USR_GRP_USR table.

4.8 Vendor Allocation - Round Robin: C1-VENALCRR

This section provides details of the Vendor Allocation - Round Robin: C1-VENALCRR algorithm.

Table 4–12 Vendor Allocation - Round Robin: C1-VENALCRR

Description	This algorithm is used for allocating cases to vendors in round-robin method.
Detailed Description	This algorithm allocates cases to vendors in round-robin method. This algorithm is invoked from the User Allocation batch (C1-USALC).
Algorithm Entity	Vendor Allocation
Program Type	java
Program Name	com.splwg.ccb.domain.collection.batch.algorithm.VendorAllocationRoundRobinAlgorithm
Parameters	NA

Table 4–12 (Cont.) Vendor Allocation - Round Robin: C1-VENALCRR

Detailed Design	This algorithm takes input as Queue code. The computation logic for case capacity is as below: ■ A1 = Total existing allocation for the vendor across all queues. ■ B1 = Total capacity of the vendor. This has to be defined in vendor on boarding screen. ■ C1 = B1 - A1 = Total available capacity of the vendor across all service types. ■ A2 = Existing allocation of the vendor for the current queue. ■ B2 = Capacity of the vendor for the queue. This is defined in queue master. ■ C2 = B2 - A2 = Total available capacity of the vendor for the current queue. ■ D1 = Available capacity for number of cases of the vendor for the queue is lower of C1 and C2. ■ A3 = Existing allocation to the vendor for a service type attached to the vendor. ■ B3 = Total capacity of the vendor for that service type. This is defined on vendor on boarding screen under section 'Associated Service Types'. If the value is blank then do not calculate capacity (C3) for that service type. ■ C3 = B3 - A3 = Total available capacity for number of cases for a vendor service type. Repeat above steps for each service type attached to the vendor. ■ Available capacity for number of cases for the vendor for a service type attached to the vendor for the queue is lower of D1 and C3. If C3 is not available for a service type then D1 should be considered as capacity. ■ Get all cases which are allocated to the queue and: - Have no vendors attached to it OR - Current allocated vendor does not have active association with the queue. ■ Get "available capacity" of cases of each vendor for each service type attached (A). ■ Get "available capacity" of OS amount of each vendor for each service type attached (B). ■ Allocate cases to vendor in a round-robin manner starting with vendor with highest available capacity of number of cases for that queue (see D1 in round-robin based capacity calculation) and then in decreasing order of capacity. ■ For every case to be allocated the system should check that case type of the case matches with case type of the service types attached with vendor. Match found: - Yes: Allocate if count of newly allocated cases for that service type and OS balance of newly allocated cases for that service type < A and B respectively. If value for B is blank then ignore validating it. - No: Move to next vendor in queue. ■ A count of freshly allocated cases should be maintained for each vendor. ■ Allocation to a particular vendor will be skipped if count of newly allocated cases for that service type or OS balance of newly allocated cases for that service type = A or B respectively. ■ All cases for which case type does not match with case type of the service types attached with any vendor in the queue will be kept allocated at queue level only. These cases should not be allocated to exception user or team. ■ If capacity of all vendors is exhausted and there are still cases pending allocation, these should be allocated to exception user or team. There will be no check for exception user's capacity. Exception user's expiry date will be checked against SC_USR_GRP_USR table.

4.9 Vendor Allocation - % Based: C1-VENALCPR

This section provides details of the Vendor Allocation - % Based: C1-VENALCPR algorithm.

Table 4–13 Vendor Allocation - % Based: C1-VENALCRR

Description	This algorithm is used for allocating cases to vendors in percentage-based method.
Detailed Description	This algorithm allocates cases to vendors in percentage-based method. This algorithm is invoked from the User Allocation batch (C1-USALC).
Algorithm Entity	Vendor Allocation
Program Type	java
Program Name	com.splwg.ccb.domain.collection.batch.algorithm.VendorAllocationPercentageBaseAlgorithm
Parameters	NA

Table 4–13 (Cont.) Vendor Allocation - % Based: C1-VENALCRR

Detailed Design	This algorithm takes input as Queue code. The computation logic for case capacity is as below: ■ A1 = Total existing allocation for the vendor across all queues. ■ B1 = Total capacity of the vendor. This has to be defined in vendor on boarding screen. ■ C1 = B1 - A1 = Total available capacity of the vendor across all service types. ■ D1 = Available capacity for no. of cases of the vendor for the queue is C1. ■ A3 = Existing allocation to the vendor for a service type attached to the vendor. ■ B3 = Total capacity of the vendor for that service type. This is defined on vendor on boarding screen under section 'Associated Service Types'. If the value is blank then do not calculate capacity (C3) for that service type. ■ C3 = B3 - A3 = Total available capacity for number of cases for a vendor service type. Repeat above steps for each service type attached to the vendor. ■ Available capacity for number of cases for the vendor, for a service type attached to the vendor for the queue is lower of D1 and C3. If C3 is not available for a service type then D1 should be considered as capacity. ■ Get all cases which are allocated to the queue and - Have no vendors attached to it OR - Current allocated vendor does not have active association with the queue. ■ Calculate % allocation for each vendor in the queue to find maximum cases of new cases that can be allocated to each vendor. ■ Get "available capacity" of cases of each vendor for each service type attached (A). ■ Get "available capacity" of OS amount of each vendor for each service type attached (B). ■ Allocate cases to vendor in a sequential manner starting with vendor with highest available capacity of number of cases for that queue (see D1 in % based capacity calculation) and then in decreasing order of capacity. ■ For every case to be allocated system should check that case type of the case matches with case type of the service types attached with vendor. Match found: -Yes: Allocate if count of newly allocated cases for that service type and OS balance of newly allocated cases for that service type < A and B respectively. If value for B is blank then ignore validating it - No: Move to next vendor in queue. ■ A count of freshly allocated cases should be maintained for each vendor. ■ Allocation to a particular vendor will be skipped if count of newly allocated cases for that service type or OS balance of newly allocated cases for that service type = A or B respectively. ■ All cases for which case type does not match with case type of the service types attached with any vendor in the queue will be kept allocated at queue level only. These cases should not be allocated to exception user or team. ■ If capacity of all vendors is exhausted and there are still cases pending allocation, these should be allocated to exception user. There will be no check for exception user's capacity. Exception user's expiry date will be checked against SC_USR_GRP_USR table.

4.10 Bulk Contact Creation: C1-BLKCNTCRE

This section provides details of the Bulk Contact Creation: C1-BLKCNTCRE algorithm.

Table 4–14 Bulk Contact Creation: C1-BLKCNTCRE

Description	This algorithm is used for allocating cases to vendors in percentage-based method.
Detailed Description	This algorithm allocates cases to vendors in percentage-based method. This algorithm is invoked from the User Allocation batch (C1-USALC).
Algorithm Entity	Bulk contact creation
Program Type	java
Program Name	com.splwg.ccb.domain.collection.batch.algorithm.BulkContactCreationAlgoComp
Parameters	NA
Detailed Design	This algorithm will be invoked from bulk contact creation batch from where the hard parameter values are set. The algorithm will call business service 'C1-GenerateCorrespondence'. addMultiple() method of 'C1-GenerateCorrespondence' will be called which in turn adds customer contact to CI_CC via add () method of the same service.

4.11 Cross Strategy Action Matrix: C1-CSAM

This section provides details of the Cross Strategy Action Matrix: C1-CSAM algorithm.

Table 4–15 Cross Strategy Action Matrix: C1-CSAM

Description	This algorithm is used for Cross Strategy Action Matrix
Detailed Description	
Algorithm Entity	Case Type- Enter status
Program Type	java
Program Name	com.splwg.ccb.domain.collection.batch.algorithm.CrossStrategyActionMatrixAlgorithm
Parameters	**Name:** CheckStatus **Required (Yes/No):** N **Description:** Y - Case types with Status N - Case types without status
Detailed Design	This algorithm will refer the CSAM admin configuration for case types and decide what action is to be taken for open case available on the entity being worked upon. It will also consider associated entity cases on the entity being worked upon. The two possible actions are: ▪ **Close the case:** Case status will be moved to next final status or the one with default switch. Business service to close the case (change case status) will be called. This action will not cure the account though. TO DO (TO DO type: C1-CSAM) will be created for the case if no final status is found for the case type or if case cannot be closed due to some other error. ▪ **Hold the case:** The business service for holding a case will be called. Hold expiry date will be set to a default value of 01-01-2100. Hold reason flag will be "CSAM". This algorithm should also get triggered during case association process.

Table 4–16 Cross Strategy Action Matrix: Sample Algorithm

Algorithm Name	C1-CSAMY
Parameters	**Name:** CheckStatus **Value:** Y

4.12 Last Payment for Account: C1-PAYDTAMTU

This section provides details of the Last Payment for Account: C1-PAYDTAMTU algorithm.

Table 4–17 Last Payment for Account: C1-PAYDTAMTU

Description	This algorithm is used to update last payment date and amount in account extension table.
Detailed Description	This algorithm will be invoked on FT freeze algorithm spot and will update Last Payment date and amount in account extension table.
Algorithm Entity	Customer class - FT Freeze
Program Type	java
Program Name	com.splwg.ccb.domain.collection.batch.algorithm.LastPaymentDtAmtUpdateAlgorithm
Parameters	NA
Detailed Design	It is invoked when the FT is freezed for payment. Algorithm will update the FT amount and FT date in Account extension table column LAST_PAYMENT_AMT and LAST_PAYMENT_DT.

4.13 Association Review Check: C1-ASORVCHK

This section provides details of the Association Review Check: C1-ASORVCHK algorithm.

Table 4–18 Association Review Check: C1-ASORVCHK

Description	This algorithm is used to check if association review is required.
Detailed Description	This is to decide if the user should review the system association of entities or not. If Association Review is Required - Stay in current status for user review. Set display date to current business date. If association Review is not required then transition to specified next status.
Algorithm Entity	Case Enter Validation
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.legal

Table 4–18 (Cont.) Association Review Check: C1-ASORVCHK

Parameters	**Name:** NextStatus **Required (Yes/No):** N **Description:** NA **Name:** AssociationReviewRequired **Required (Yes/No):** Y **Description:** NA
Detailed Design	It is invoked in the pending status of Legal Process. It decides whether the user should review the system association of entities or not. 'Y' in the algorithm parameter specifies that Association review is required.

Table 4–19 Association Review Check: Sample Algorithm

Algorithm Name	C1-ASORVCHK
Parameters	**Name:** NextStatus **Value:** ASSNEWLSP **Name:** AssociationReviewRequired **Value:** Y

4.14 Validate Expired Default Notice: C1-DEFNOEXP

This section provides details of the Validate Expired Default Notice: C1-DEFNOEXP algorithm.

Table 4–20 Validate Expired Default Notice: C1-DEFNOEXP

Description	This algorithm is used to validate expired default notices.
Detailed Description	This algorithm returns an error if there is no default notice on a given account or a default notice has not yet expired.
Algorithm Entity	Case Type - Enter Validation
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.legal

Table 4–20 (Cont.) Validate Expired Default Notice: C1-DEFNOEXP

Parameters	**Name:** associationType **Required (Yes/No):** Y **Description:** NA **Name:** validationfailureOption **Required (Yes/No):** Y **Description:** NA **Name:** toDoType **Required (Yes/No):** N **Description:** NA **Name:** toDoRole **Required (Yes/No):** N **Description:** NA
Detailed Design	It is invoked in the pending status of the Legal Process case. It checks if the default notice has expired for a particular account.

Table 4–21 Validate Expired Default Notice: Sample Algorithm

Algorithm Name	C1-DEFNOEXP
Parameters	**Name:** associationType **Value:** P **Name:** validationfailureOption **Value:** N **Name:** toDoType **Value:** C1-TD-DN **Name:** toDoRole **Value:**

4.15 Associate Related Entity: C1-ASSOENTY

This section provides details of the Associate Related Entity: C1-ASSOENTY algorithm.

Table 4–22 Associate Related Entity: C1-ASSOENTY

Description	This algorithm is used to associate related entities with the case.
Detailed Description	This algorithm pulls the related entities associated with the case.
Algorithm Entity	Case Type - Enter Validation

Table 4–22 (Cont.) Associate Related Entity: C1-ASSOENTY

Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.legal
Parameters	**Name:** hostId **Required (Yes/No):** Y **Description:** NA **Name:** toDoType **Required (Yes/No):** Y **Description:** NA **Name:** toDoRole **Required (Yes/No):** N **Description:** NA
Detailed Design	It is invoked in the pending state of the Legal Case process. The algorithm associates the primary account with the persons attached to it and also the accounts which have the same set of financially responsible customers as in the primary account.

Table 4–23 Associate Related Entity: Sample Algorithm

Algorithm Name	C1-ASSOENTY
Parameters	**Name:** hostId **Value:** NGP **Name:** toDoType **Value:** C1-TD-AC **Name:** toDoRole **Value:**

4.16 Validate Legal Case Exists: C1-CHKLGL

This section provides details of the Validate Legal Case Exists: C1-CHKLGL algorithm.

Table 4–24 Validate Legal Case Exists: C1-CHKLGL

Description	This algorithm is used to validate if an active legal case exists at the same time.
Detailed Description	This algorithm checks if a legal case is already running on the primary account any account in the collection with the same owner.
Algorithm Entity	Case Enter Validation
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.legal

Table 4–24 (Cont.) Validate Legal Case Exists: C1-CHKLGL

Parameters	**Name:** Case Category **Required (Yes/No):** Y **Description:** NA **Name:** toDoType **Required (Yes/No):** Y **Description:** NA **Name:** toDoRole **Required (Yes/No):** N **Description:** NA
Detailed Design	It is invoked in the pending state of the Legal Process case. It checks if there is any legal case running on the primary account or its related entities.

Table 4–25 Validate Legal Case Exists: Sample Algorithm

Algorithm Name	C1-ASSOENTY
Parameters	**Name:** Case Category **Value:** LEGL **Name:** toDoType **Value:** C1-TD-CL **Name:** toDoRole **Value:**

4.17 Assign New LSP: C1-ASGNLSP

This section provides details of the Assign New LSP: C1-ASGNLSP algorithm.

Table 4–26 Assign New LSP: C1-ASGNLSP

Description	This algorithm is used to assign LSP to the case.
Detailed Description	This algorithm assigns the LSP to the case either automatically or let the user assign manually depending on the value entered in the algorithm parameters.
Algorithm Entity	Case Enter Status
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.legal

Table 4–26 (Cont.) Assign New LSP: C1-ASGNLSP

Parameters	**Name:** New Allocation And Review Option **Required (Yes/No):** N **Description:** NA **Name:** Change LSP Allocation Option **Required (Yes/No):** N **Description:** NA **Name:** Reset Document Submission Date **Required (Yes/No):** N **Description:** NA **Name:** Previous Allocation Check **Required (Yes/No):** N **Description:** NA **Name:** Next Status **Required (Yes/No):** N **Description:** NA
Detailed Design	It is invoked in the Assign New LSP status of the Legal Process case. Depending on the different algorithm parameter values, the LSP is assigned automatically or manually (both in cases of First time assignment or change assignment).

Table 4–27 Assign New LSP: Sample Algorithm

Algorithm Name	C1-ASGNLSP
Parameters	**Name:** New Allocation And Review Option **Value:** AUTO_WITH_REVIEW_PRVALLOC **Name:** Change LSP Allocation Option **Value:** AUTO_WITH_REVIEW **Name:** Reset Document Submission Date **Value:** N **Name:** Previous Allocation Check **Value:** Y **Name:** Next Status **Value:** PREPLGLDOC

4.18 Check Approval Requirement: C1-APPRCHK

This section provides details of the Check Approval Requirement: C1-APPRCHK algorithm.

Table 4–28 Check Approval Requirement: C1-APPRCHK

Description	This algorithm is used to check the need of approval.
Detailed Description	This algorithm checks if LSP assignments should be approved.
Algorithm Entity	Case Type - Enter Processing
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.legal
Parameters	**Name:** Exposure Threshold **Required (Yes/No):** N **Description:** NA **Name:** Approval Request Status **Required (Yes/No):** N **Description:** NA **Name:** approvedStatus **Required (Yes/No):** N **Description:** NA **Name:** rejectRequestStatus **Required (Yes/No):** N **Description:** NA
Detailed Design	It is invoked in the Prepare Legal Documents status of the Legal Process Case. It checks if the approval is required for the LSP assignment depending on the algorithm parameter values. It also decides where to transit the case.

Table 4–29 Check Approval Requirement: Sample Algorithm

Algorithm Name	C1-ASGNLSP
Parameters	**Name:** Exposure Threshold **Value:** 10 **Name:** Approval Request Status **Value:** PENDINGAPP **Name:** approvedStatus **Value:** WTFRLSPACK **Name:** rejectRequestStatus **Value:** ASSNEWLSP

4.19 Save the Status Before Change LSP: C1-SAVESTATUS

This section provides details of the Save the Status Before Change LSP: C1-SAVESTATUS algorithm.

Table 4–30 Save the Status Before Change LSP: C1-SAVESTATUS

Description	This algorithm is used to save the status before the status changes in LSP.
Detailed Description	This algorithm saves the status from where it came to Change LSP status. This will be stored in CI_LSP_DTLS table.
Algorithm Entity	Case Type-Enter Processing
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.legal
Parameters	NA
Detailed Design	It is invoked in the Change or Retire LSP status of the Legal Process Case. It stores the previous state of the case so that it returns to that state after the LSP for the case is changed.

4.20 Resume Status from Previous LSP: C1-RESSTATUS

This section provides details of the Resume Status from Previous LSP: C1-RESSTATUS algorithm.

Table 4–31 Resume Status from Previous LSP: C1-RESSTATUS

Description	This algorithm is used to resume status from previous LSP.
Detailed Description	This algorithm resumes the previous state stored while changing LSP.
Algorithm Entity	Customer class - FT Freeze
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.legal
Parameters	NA
Detailed Design	It is invoked in the Legal in Progress status of the Legal Process Case. It resumes the status where the case was previously in before changing the LSP for the case.

4.21 Check Submission Date: C1-CHKSUBDT1

This section provides details of the Check Submission Date: C1-CHKSUBDT1 algorithm.

Table 4–32 Check Submission Date: C1-CHKSUBDT1

Description	This algorithm is used to check submission date.
Detailed Description	This algorithm checks if the document submission date is filled from screen. If it is present, the case is auto transitioned to 'WAIT FOR LSP ACKNOWLEDGMENT' status directly from 'ASSIGN NEW LSP' status.
Algorithm Entity	Case Auto Transition Validation

Table 4–32 (Cont.) Check Submission Date: C1-CHKSUBDT1

Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.legal
Parameters	**Name:** nextStatus **Required (Yes/No):** Y **Description:** NA **Name:** changeStatus **Required (Yes/No):** Y **Description:** NA
Detailed Design	It is invoked in the Prepare Legal Documents status of the Legal Process case. This algorithm checks for the presence of document submission date in the database. If document submission date is present in the database, then based on the soft parameter it will transition the case to next status.

Table 4–33 Check Submission Date: Sample Algorithm

Algorithm Name	C1-CHKSUBDT1
Parameters	**Name:** nextStatus **Value:** WTFRLSPACK **Name:** changeStatus **Value:** Y

4.22 Update LSP (CLOS): C1-LSPSTATUS

This section provides details of the Update LSP (CLOS): C1-LSPSTATUS algorithm.

Table 4–34 Update LSP (CLOS): C1-LSPSTATUS

Description	Legal Proceedings - Update Status
Detailed Description	This algorithm updates the end date and assignment status of the CI_LSP_DTLS table after the Legal case is either closed or cancelled.
Algorithm Entity	Case Type-Enter Status
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.legal
Parameters	**Name:** lspAssignmentStatus **Required (Yes/No):** Y **Description:** NA
Detailed Design	It is invoked in the Complete, Withdraw status of the Legal Process case. This algorithm updates the end date and assignment status of the CI_LSP_DTLS table after the Legal case is either completed or withdrawn.

Table 4–35 Update LSP (CLOS): Sample Algorithm

Algorithm Name	C1-LSPSTATUS
Parameters	**Name:** lspAssignmentStatus **Value:** CLOS

4.23 Update LSP (CANCEL): C1-LSPSTACAN

This section provides details of the Update LSP (CANCEL): C1-LSPSTACAN algorithm.

Table 4–36 Update LSP (CANCEL): C1-LSPSTACAN

Description	Legal Proceedings - Update Status
Detailed Description	This algorithm updates the end date and assignment status of the CI_LSP_DTLS table after the Legal case is either closed or cancelled.
Algorithm Entity	Case Type-Enter Status
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.legal
Parameters	**Name:** lspAssignmentStatus **Required (Yes/No):** Y **Description:** NA
Detailed Design	It is invoked in the CANCEL status of the Legal Process case. This algorithm updates the end date and assignment status of the CI_LSP_DTLS table after the Legal case is cancelled.

Table 4–37 Update LSP (CANCEL): Sample Algorithm

Algorithm Name	C1-LSPSTACAN
Parameters	**Name:** lspAssignmentStatus **Value:** CAN

4.24 Validate Expired Default Notice: C1-DEFNOTEXP

This section provides details of the Validate Expired Default Notice: C1-DEFNOTEXP algorithm.

Table 4–38 Validate Expired Default Notice: C1-DEFNOTEXP

Description	Validate Expired Default Notice
Detailed Description	This algorithm returns an error if there is no default notice on a given account or a default notice has not yet expired.
Algorithm Entity	Case Type - Enter Validation
Program Type	java

Table 4–38 (Cont.) Validate Expired Default Notice: C1-DEFNOTEXP

Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.legal
Parameters	**Name:** associationType **Required (Yes/No):** Y **Description:** NA **Name:** validationfailureOption **Required (Yes/No):** Y **Description:** NA **Name:** toDoType **Required (Yes/No):** N **Description:** NA **Name:** toDoRole **Required (Yes/No):** N **Description:** NA
Detailed Design	It is invoked in the Assign New LSP status of the Legal Process case. It checks if the default notice has expired for a particular account.

Table 4–39 Validate Expired Default Notice: Sample Algorithm

Algorithm Name	C1-DEFNOTEXP
Parameters	**Name:** associationType **Value:** A **Name:** validationfailureOption **Value:** Y **Name:** toDoType **Value:** C1-TD-DN **Name:** toDoRole **Value:**

4.25 Collateral Verification: C1-VRFYCOLS

This section provides details of the Collateral Verification: C1-VRFYCOLS algorithm.

Table 4–40 Collateral Verification: C1-VRFYCOLS

Description	Collateral Verification
Detailed Description	This will perform following validations for the collateral with the case: ■ If the soft parameter for Collateral type to this algorithm type is "PROPERTY", then one collateral is associated with the case and that Collateral is associated with Facility for the primary account associated with the case. ■ If collateral type soft parameter is blank, then above validation should be ignored and Collateral status is set to Not Sold. ■ It will also validate that if there is no active Asset repossession case running for the collateral. If any of the above validations fail, case creation process should be terminated.
Algorithm Entity	Case Type-Enter Status
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.AssetRepo.CollateralVerification
Parameters	**Name:** Collateral Type **Required (Yes/No):** N **Description:** NA
Detailed Design	It is invoked in the Pending status of the Asset Repossession Process case. It Verifies the collateral associated with account.

Table 4–41 Collateral Verification: Sample Algorithm

Algorithm Name	C1-VRFYCOLS
Parameters	**Name:** Collateral Type **Value:** PROPERTY

4.26 Account Association for Asset Repossession Case: C1-ARSACCTS

This section provides details of the Account Association for Asset Repossession Case: C1-ARSACCTS algorithm.

Table 4–42 Account Association for Asset Repossession Case: C1-ARSACCTS

Description	Account Association for Asset repossession case
Detailed Description	This algorithm will perform following actions: ■ It gets all facilities to which this collateral is associated and all accounts for these facilities. ■ It associates these accounts with the case. Scope of this association is limited to accounts already in collections. This process will not check for any accounts not in collections. This algorithm doesn't have any soft parameter.
Algorithm Entity	Case Type-Enter Status

Table 4–42 (Cont.) Account Association for Asset Repossession Case: C1-ARSACCTS

Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.AssetRepo.AccountAssociationForAssetRepossessionCase
Parameters	NA
Detailed Design	It is invoked in the Pending status of the Asset Repossession Process case. It will associate facilities of account with case.

4.27 Customer Association for Asset Repossession Case: C1-ARSCUSTS

This section provides details of the Customer Association for Asset Repossession Case: C1-ARSCUSTS algorithm.

Table 4–43 Customer Association for Asset Repossession Case: C1-ARSCUSTS

Description	Customer Association for Asset repossession case
Detailed Description	This algorithm performs the following actions: ■ It gets all customers who are the owners for the selected collateral ■ It associates these customers with the case Scope of this association is limited to customers already in collections. This process will not check for any customers not in collections. This algorithm does not have any soft parameter.
Algorithm Entity	Case Type-Enter Status
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.AssetRepo.CustomerAssociationForAssetRepossessionCase
Parameters	NA
Detailed Design	It is invoked in the Pending status of the Asset Repossession Process case. It will associate facilities of customer with case.

4.28 Update Collateral Property: C1-SETCOLLPR

This section provides details of the Update Collateral Property: C1-SETCOLLPR algorithm.

Table 4–44 Update Collateral Property: C1-SETCOLLPR

Description	Update Collateral Property
Detailed Description	This algorithm will perform following operations: ■ If the value of updateCollateralProperty soft parameter is "SET" and type of possession is "Warrant" then Fetch the collateral for which case is created and update the IS_LEGAL_SW= "Y" and populate the case_ID on this collateral. ■ If the value of updateCollateralProperty soft parameter is "RESET" then Fetch the collateral for which case is created and update the IS_LEGAL_SW= "N" and IS_REPO_SW= "N" nullify the case_ID on this collateral.
Algorithm Entity	Case Type-Enter Status

Table 4–44 (Cont.) Update Collateral Property: C1-SETCOLLPR

Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.AssetRepo.UpdateCollateralProperty
Parameters	**Name:** UpdateCollateralProperty **Required (Yes/No):** Y **Description:** NA
Detailed Design	It is invoked in the Pending status of the Asset Repossession Process case. It updates the collateral Properties like IS_LEGAL_SW, IS_REPO_SW depending on user inputs.

4.29 Close To do's Algorithm: C1-CLSTODOA

This section provides details of the Close To do's Algorithm: C1-CLSTODOA algorithm.

Table 4–45 Close To do's Algorithm: C1-CLSTODOA

Description	Close To do's algorithm
Detailed Description	This process will close all To-Do's of specific To-do types associated with the case. Up to five To-Do types can be given to this algorithm to close.
Algorithm Entity	Case Type-Exit Status
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.CloseTodo

Table 4–45 (Cont.) Close To do's Algorithm: C1-CLSTODOA

Parameters	**Name:** To Do Type1 **Required (Yes/No):** N **Description:** NA **Name:** To Do Type2 **Required (Yes/No):** N **Description:** NA **Name:** To Do Type3 **Required (Yes/No):** N **Description:** NA **Name:** To Do Type4 **Required (Yes/No):** N **Description:** NA **Name:** To Do Type5 **Required (Yes/No):** N **Description:** NA
Detailed Design	It is invoked while exiting from Pending status of the Asset Repossession Process case. This process will close all To-Do's of "No activity" To-do types associated with the case.

Table 4–46 Close To do's Algorithm: Sample Algorithm

Algorithm Name	C1-ARSCUSTS
Parameters	**Name:** To Do Type1 **Value:** C1-ANA1 **Name:** To Do Type2 **Value:** C1-ANA2 **Name:** To Do Type3 **Value:** **Name:** To Do Type4 **Value:** **Name:** To Do Type5 **Value:**

4.30 Validations for Mandatory Characteristics: C1-CHARVALU

This section provides details of the Validations for Mandatory Characteristics: C1-CHARVALU algorithm.

Table 4–47 Validations for Mandatory Characteristics: C1-CHARVALU

Description	Validations for Mandatory Characteristics
Detailed Description	Subjective Validations for Mandatory Characteristics
Algorithm Entity	Case Type-Enter Status
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.AssetRepo.MandatoryCharacteristics
Parameters	**Name:** ReferenceCharacteristicsValue **Required (Yes/No):** Y **Description:** NA **Name:** ReferenceCharacteristic **Required (Yes/No):** Y **Description:** NA **Name:** CaseCharacteristics1 **Required (Yes/No):** N **Description:** NA **Name:** CaseCharacteristics2 **Required (Yes/No):** N **Description:** NA **Name:** CaseCharacteristics3 **Required (Yes/No):** N **Description:** NA **Name:** CaseCharacteristics4 **Required (Yes/No):** N **Description:** NA **Name:** CaseCharacteristics5 **Required (Yes/No):** N **Description:** NA
Detailed Design	It is invoked in Effected Possession status of the Asset Repossession Process case. This algorithm will carry out subjective validation based on the type of input.

Table 4–48 Validations for Mandatory Characteristics: Sample Algorithm

Algorithm Name	C1-CHARVALU
Parameters	**Name:** ReferenceCharacteristicsValue **Value:** Type of Possession **Name:** ReferenceCharacteristic **Value:** Voluntary Possession **Name:** CaseCharacteristics1 **Value:** Vacancy Date **Name:** CaseCharacteristics2 **Value:** Vacancy Possession Indemnity Policy Reference **Name:** CaseCharacteristics3 **Value:** Property Surrender Letter Reference **Name:** CaseCharacteristics4 **Value:** Property Surrender Letter Reference **Name:** CaseCharacteristics5 **Value:**

4.31 Validations for Mandatory Characteristics: C1-CHARVALA

This section provides details of the Validations for Mandatory Characteristics: C1-CHARVALA algorithm.

Table 4–49 Validations for Mandatory Characteristics: C1-CHARVALA

Description	Validations for Mandatory Characteristics
Detailed Description	Subjective Validations for Mandatory Characteristics
Algorithm Entity	Case Type-Exit Status
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.AssetRepo.MandatoryCharacteristics

Table 4–49 (Cont.) Validations for Mandatory Characteristics: C1-CHARVALA

Parameters	**Name:** ReferenceCharacteristicsValue **Required (Yes/No):** Y **Description:** NA **Name:** ReferenceCharacteristic **Required (Yes/No):** Y **Description:** NA **Name:** CaseCharacteristics1 **Required (Yes/No):** N **Description:** NA **Name:** CaseCharacteristics2 **Required (Yes/No):** N **Description:** NA **Name:** CaseCharacteristics3 **Required (Yes/No):** N **Description:** NA **Name:** CaseCharacteristics4 **Required (Yes/No):** N **Description:** NA **Name:** CaseCharacteristics5 **Required (Yes/No):** N **Description:** NA
Detailed Design	It is invoked in Effected Possession status of the Asset Repossession Process case. This algorithm will carry out subjective validation based on the type of input.

Table 4–50 Validations for Mandatory Characteristics: Sample Algorithm

Algorithm Name	C1-CHARVALU
Parameters	**Name:** ReferenceCharacteristicsValue **Value:** Type of Possession **Name:** ReferenceCharacteristic **Value:** Voluntary Possession **Name:** CaseCharacteristics1 **Value:** Legal Case ID **Name:** CaseCharacteristics2 **Value:** **Name:** CaseCharacteristics3 **Value:** **Name:** CaseCharacteristics4 **Value:** **Name:** CaseCharacteristics5 **Value:**

4.32 Update Collateral Status in the Host: C1-UPCOLLSTZ

This section provides details of the Update Collateral Status in the Host: C1-UPCOLLSTZ algorithm.

Table 4–51 Update Collateral Status in the Host: C1-UPCOLLSTZ

Description	Update Collateral Status in the host
Detailed Description	This process updates the collateral status in the host. The value of status to be set will be passed as parameter to the process. If the update fails for any reason, system should create a To-do. Message for the To-do should be configured based on type of update which failed. To-do should be assigned to the To-do Role set as parameter to this process. If the parameter is left blank, To-do should be assigned to the default role.
Algorithm Entity	Case Type-Enter Status
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.AssetRepo.UpdateCollateralStatusInTheHost

Table 4–51 (Cont.) Update Collateral Status in the Host: C1-UPCOLLSTZ

Parameters	**Name:** To Do Role **Required (Yes/No):** N **Description:** NA **Name:** To Do Type **Required (Yes/No):** Y **Description:** NA **Name:** Collateral Status **Required (Yes/No):** Y **Description:** NA
Detailed Design	It is invoked in Effected Possession status of the Asset Repossession Process case. This process will update the collateral status in the host.

Table 4–52 Update Collateral Status in the Host: Sample Algorithm

Algorithm Name	C1-UPCOLLSTZ
Parameters	**Name:** To Do Role **Value:** **Name:** To Do Type **Value:** C1-TD-UC **Name:** Collateral Status **Value:** Sold

4.33 Initiate Collateral Valuation: C1-COLLVALX

This section provides details of the Initiate Collateral Valuation: C1-COLLVALX algorithm.

Table 4–53 Initiate Collateral Valuation: C1-COLLVALX

Description	Update Collateral Status in the host
Detailed Description	This algorithm works as follows: System should check if "X" days have elapsed since the last assessment was done for the collateral. That is check if (Assessment date + X) <= Current business date. Number of days, X, will be set as Assessment Expiry Days parameter for this process. If yes - Create a To-do to alert the user that collateral valuation is required. This to-do should be associated with the case. To-do Type is passed as a parameter to the process. However, To-do should not be created if: ■ A To-do of same to-do type is already open for the case ■ A To-do of same to-do type was closed within past "Y" days To-do should be assigned to the To-do Role set as parameter to this process. If the parameter is left blank, To-do should be assigned to the default role.
Algorithm Entity	Case Type-Enter Status
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.AssetRepo.UpdateCollateralStatusInTheHost
Parameters	**Name:** To Do Role **Required (Yes/No):** N **Description:** NA **Name:** To Do Type **Required (Yes/No):** Y **Description:** NA **Name:** Days Since Closure Of Last To Do **Required (Yes/No):** Y **Description:** NA **Name:** Assessment Expiry Days **Required (Yes/No):** Y **Description:** NA
Detailed Design	It is invoked while exiting from Pending status of the Asset Repossession Process case. This process will close all To-Do's of "Asset repossession No activity" To-do types associated with the case.

Table 4–54 Initiate Collateral Valuation: Sample Algorithm

Algorithm Name	C1-COLLVALX
Parameters	**Name:** To Do Role **Value:** C1-ASSETRE **Name:** To Do Type **Value:** C1-TD-UC **Name:** Days Since Closure Of Last To Do **Value:** 5 **Name:** Assessment Expiry Days **Value:** 5

4.34 Close To do's Algorithm: C1-CLSTODOV

This section provides details of the Close To do's Algorithm: C1-CLSTODOV algorithm.

Table 4–55 Close To do's Algorithm: C1-CLSTODOV

Description	Close To do's algorithm
Detailed Description	This process will close all To-Do's of specific To-do types associated with the case. Up to five To-Do types can be given to this algorithm to close.
Algorithm Entity	Case Type-Exit Status
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.CloseTodo

Table 4–55 (Cont.) Close To do's Algorithm: C1-CLSTODOV

Parameters	**Name:** To Do Type1 **Required (Yes/No):** N **Description:** NA **Name:** To Do Type2 **Required (Yes/No):** N **Description:** NA **Name:** To Do Type3 **Required (Yes/No):** N **Description:** NA **Name:** To Do Type4 **Required (Yes/No):** N **Description:** NA **Name:** To Do Type5 **Required (Yes/No):** N **Description:** NA
Detailed Design	It is invoked while exiting from Sale In-Progress status of the Asset Repossession Process case. This process will close all To-Do's of "No activity" To-do types associated with the case.

Table 4–56 Close To do's Algorithm: Sample Algorithm

Algorithm Name	C1-CLSTODOV
Parameters	**Name:** To Do Type1 **Value:** C1-LNA1 **Name:** To Do Type2 **Value:** C1-LNA1 **Name:** To Do Type3 **Value:** C1-TD-CV **Name:** To Do Type4 **Value:** **Name:** To Do Type5 **Value:**

4.35 Validations for Mandatory Characteristics: C1-CHARVALC

This section provides details of the Validations for Mandatory Characteristics: C1-CHARVALC algorithm.

Table 4–57 Validations for Mandatory Characteristics: C1-CHARVALC

Description	Validations for Mandatory Characteristics
Detailed Description	Subjective Validations for Mandatory Characteristics
Algorithm Entity	Case Type-Enter Status
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.AssetRepo.MandatoryCharacteristics
Parameters	**Name:** ReferenceCharacteristicsValue **Required (Yes/No):** Y **Description:** NA **Name:** ReferenceCharacteristic **Required (Yes/No):** Y **Description:** NA **Name:** CaseCharacteristics1 **Required (Yes/No):** N **Description:** NA **Name:** CaseCharacteristics2 **Required (Yes/No):** N **Description:** NA **Name:** CaseCharacteristics3 **Required (Yes/No):** N **Description:** NA **Name:** CaseCharacteristics4 **Required (Yes/No):** N **Description:** NA **Name:** CaseCharacteristics5 **Required (Yes/No):** N **Description:** NA
Detailed Design	It is invoked in Settlement status of the Asset Repossession Process case. This algorithm will carry out subjective validation based on the type of input.

Table 4–58 Validations for Mandatory Characteristics: Sample Algorithm

Algorithm Name	C1-CHARVALU
Parameters	**Name:** ReferenceCharacteristicsValue **Value:** Type of Possession **Name:** ReferenceCharacteristic **Value:** Voluntary Possession **Name:** CaseCharacteristics1 **Value:** Contactor Details **Name:** CaseCharacteristics2 **Value:** Conveyance Details **Name:** CaseCharacteristics3 **Value:** **Name:** CaseCharacteristics4 **Value:** **Name:** CaseCharacteristics5 **Value:**

4.36 Update Collateral Status in the Host: C1-UPCOLLSTX

This section provides details of the Update Collateral Status in the Host: C1-UPCOLLSTX algorithm.

Table 4–59 Update Collateral Status in the Host: C1-UPCOLLSTX

Description	Update Collateral Status in the host
Detailed Description	This process updates the collateral status in the host. The value of status to be set will be passed as parameter to the process. If the update fails for any reason, system should create a To-do. Message for the To-do should be configured based on type of update which failed. To-do should be assigned to the To-do Role set as parameter to this process. If the parameter is left blank, To-do should be assigned to the default role.
Algorithm Entity	Case Type-Enter Status
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.AssetRepo.UpdateCollateralStatusInTheHost

Table 4–59 (Cont.) Update Collateral Status in the Host: C1-UPCOLLSTX

Parameters	**Name:** To Do Role **Required (Yes/No):** N **Description:** NA **Name:** To Do Type **Required (Yes/No):** Y **Description:** NA **Name:** Collateral Status **Required (Yes/No):** Y **Description:** NA
Detailed Design	It is invoked in Settlement status of the Asset Repossession Process case. This process will update the collateral status in the host.

Table 4–60 Update Collateral Status in the Host: Sample Algorithm

Algorithm Name	C1-UPCOLLSTZ
Parameters	**Name:** To Do Role **Value:** **Name:** To Do Type **Value:** C1-TD-UC **Name:** Collateral Status **Value:** Sold

4.37 Validation Settlement: C1-VALSET

This section provides details of the Validation Settlement: C1-VALSET algorithm.

Table 4–61 Validation Settlement: C1-VALSET

Description	Validation Settlement
Detailed Description	This algorithm will perform following actions: Before completing the asset repossession case, the below validations should be done for the case: ■ Collateral should have a settlement date ■ Realization status for the collateral should be "Complete" Transition to completed status will fail if above validations fail.
Algorithm Entity	Case Type-Exit Status
Program Type	java

Table 4–61 (Cont.) Validation Settlement: C1-VALSET

Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.AssetRepo.ValidateCollateralSettlementStatus
Parameters	**Name:** Realization Status **Required (Yes/No):** Y **Description:** NA
Detailed Design	It is invoked in Settlement status of the Asset Repossession Process case. This process will update the collateral status in the host.

Table 4–62 Validation Settlement: Sample Algorithm

Algorithm Name	C1-UPCOLLSTZ
Parameters	**Name:** Realization Status **Value:** REALIZATION_COMPLETE

4.38 Initiate LMI Process: C1-INITLMI

This section provides details of the Initiate LMI Process: C1-INITLMI algorithm.

Table 4–63 Initiate LMI Process: C1-INITLMI

Description	Initiate LMI Process
Detailed Description	Parameters to the algorithm must be as follows: ■ For Initiate LMI Options: 1) "Initiate LMI with highest insured amount" use HIGH_INSUR_AMT 2) "Initiate LMI from a specific insurer first" use SPEC_INSURER ■ For No LMI Option: 1) "Mark primary account for strategy review" use PRIMARY 2) "Mark all accounts for strategy review" use ALL 3) "No Action" use NA
Algorithm Entity	Case Type-Exit Status
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.AssetRepo.InitiateLMIP

Table 4–63 (Cont.) Initiate LMI Process: C1-INITLMI

Parameters	**Name:** Balance Threshold **Required (Yes/No):** Y **Description:** NA **Name:** LMI Case Type **Required (Yes/No):** Y **Description:** NA **Name:** Initiate LMI Options **Required (Yes/No):** Y **Description:** NA **Name:** LMI Insurer Code **Required (Yes/No):** Y **Description:** NA **Name:** No LMI Option **Required (Yes/No):** Y **Description:** NA
Detailed Design	It is invoked in Settlement status of the Asset Repossession Process case. This process will validate realization status and settlement date for collateral.

Table 4–64 Initiate LMI Process: Sample Algorithm

Algorithm Name	C1-INITLMI
Parameters	**Name:** Balance Threshold **Value:** 1000 **Name:** LMI Case Type **Value:** C1_LMI **Name:** Initiate LMI Options **Value:** HIGH_INSUR_AMT **Name:** LMI Insurer Code **Value:** QBE **Name:** No LMI Option **Value:** ALL

4.39 Close To do's Algorithm: C1-CLSTODO

This section provides details of the Close To do's Algorithm: C1-CLSTODO algorithm.

Table 4–65 Close To do's Algorithm: C1-CLSTODO

Description	Close To do's algorithm
Detailed Description	This process will close all To-Do's of specific To-do types associated with the case. Up to five To-Do types can be given to this algorithm to close.
Algorithm Entity	Case Type-Exit Status
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.CloseTodo
Parameters	**Name:** To Do Type1 **Required (Yes/No):** N **Description:** NA **Name:** To Do Type2 **Required (Yes/No):** N **Description:** NA **Name:** To Do Type3 **Required (Yes/No):** N **Description:** NA **Name:** To Do Type4 **Required (Yes/No):** N **Description:** NA **Name:** To Do Type5 **Required (Yes/No):** N **Description:** NA
Detailed Design	It is invoked while exiting from Settlement status of the Asset Repossession Process case. This process will close all To-Do's associated with the case.

Table 4–66 Close To do's Algorithm: Sample Algorithm

Algorithm Name	C1-CLSTODO
Parameters	**Name:** To Do Type1 **Value:** C1-TD-CL **Name:** To Do Type2 **Value:** C1-TD-AC **Name:** To Do Type3 **Value:** C1-TD-DN **Name:** To Do Type4 **Value:** C1-DNA1 **Name:** To Do Type5 **Value:**

4.40 Update Collateral Property: C1-RESETCOLL

This section provides details of the Update Collateral Property: C1-RESETCOLL algorithm.

Table 4–67 Update Collateral Property: C1-RESETCOLL

Description	Update Collateral Property
Detailed Description	This algorithm will perform following operations: ▪ If the value of updateCollateralProperty soft parameter is "SET" and type of possession is "Warrant" then Fetch the collateral for which case is created and update the IS_LEGAL_SW= "Y" and populate the case_ID on this collateral. ▪ If the value of updateCollateralProperty soft parameter is "RESET" then Fetch the collateral for which case is created and update the IS_LEGAL_SW= "N" and IS_REPO_SW= "N" nullify the case_ID on this collateral.
Algorithm Entity	Case Type-Enter Status
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.AssetRepo.UpdateCollateralProperty
Parameters	**Name:** UpdateCollateralProperty **Required (Yes/No):** Y **Description:** NA
Detailed Design	It is invoked in the Cancelled status of the Asset Repossession Process case. It will update the collateral Properties like IS_LEGAL_SW, IS_REPO_SW depending upon user inputs.

Table 4–68 Update Collateral Property: Sample Algorithm

Algorithm Name	C1-RESETCOLL
Parameters	**Name:** UpdateCollateralProperty **Value:** RESET

4.41 Update Collateral Status in the Host: C1-UPCOLLSTY

This section provides details of the Update Collateral Status in the Host: C1-UPCOLLSTY algorithm.

Table 4–69 Update Collateral Status in the Host: C1-UPCOLLSTY

Description	Update Collateral Status in the host
Detailed Description	This process will update the collateral status in the host. The value of status to be set will be passed as parameter to the process. If the update fails for any reason, system should create a To-do. Message for the To-do should be configured based on type of update which failed. To-do should be assigned to the To-do Role set as parameter to this process. If the parameter is left blank, To-do should be assigned to the default role.
Algorithm Entity	Case Type-Enter Status
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.AssetRepo.UpdateCollateralStatusInTheHost
Parameters	**Name:** To Do Role **Required (Yes/No):** N **Description:** NA **Name:** To Do Type **Required (Yes/No):** Y **Description:** NA **Name:** Collateral Status **Required (Yes/No):** Y **Description:** NA
Detailed Design	It is invoked in Withdrawn status of the Asset Repossession Process case. This process will update the collateral status in the host.

Table 4–70 Update Collateral Status in the Host: Sample Algorithm

Algorithm Name	C1-UPCOLLSTY
Parameters	**Name:** To Do Role **Value:** **Name:** To Do Type **Value:** C1-TD-UC **Name:** Collateral Status **Value:** With the Customer

4.42 PTP Active Algorithm: C1-PTPACTIVE

This section provides details of the PTP Active Algorithm: C1-PTPACTIVE algorithm.

Table 4–71 PTP Active Algorithm: C1-PTPACTIVE

Description	Algorithm to generate letter or SMS on Active Status
Detailed Description	This algorithm is used to generate letter or SMS when PTP moves to Active state.
Algorithm Entity	PTP Active Algorithm
Program Type	java
Program Name	com.splwg.ccb.domain.customerinfo.paymentPlan.CollectionPTPActiveForNgpAlgor ithm

Table 4–71 (Cont.) PTP Active Algorithm: C1-PTPACTIVE

Parameters	**Name:** contactTypeForLetter **Required (Yes/No):** **Description:** Contact Type for Letter generation **Name:** contactClassForLetter **Required (Yes/No):** **Description:** Contact Class for letter generation **Name:** contactMethodForLetter **Required (Yes/No):** **Description:** Contact Method for Letter generation **Name:** contactTypeForSMS **Required (Yes/No):** **Description:** Contact Type for SMS **Name:** contactClassForSMS **Required (Yes/No):** **Description:** Contact Class for SMS **Name:** contactMethodForSMS **Required (Yes/No):** **Description:** Contact Method for SMS
Detailed Design	This algorithm invokes **GenerateContactForPTP** service which creates the contact (generate Letter or SMS) when PTP moves to Active state.

Table 4–72 PTP Active Algorithm: Sample Algorithm

Algorithm Name	C1-PTPACTIVE
Parameters	**Name:** contactTypeForLetter **Value:** OVERDUE **Name:** contactClassForLetter **Value:** CCC **Name:** contactMethodForLetter **Value:** OTBL **Name:** contactTypeForSMS **Value:** OVERDUE **Name:** contactClassForSMS **Value:** CCC **Name:** contactMethodForSMS **Value:** OTBS

4.43 PTP Active Algorithm: C1-PTPKEPT

This section provides details of the PTP Active Algorithm: C1-PTPKEPT algorithm.

Table 4–73 PTP Active Algorithm: C1-PTPKEPT

Description	Algorithm to generate letter or SMS on **Kept** status.
Detailed Description	This algorithm is used to generate letter or SMS when PTP moves to Kept state.
Algorithm Entity	PTP Kept Algorithm
Program Type	java

Table 4–73 (Cont.) PTP Active Algorithm: C1-PTPKEPT

Program Name	com.splwg.ccb.domain.customerinfo.paymentPlan.CollectionPTPKeptForNgpAlgorithm
Parameters	**Name:** contactTypeForLetter **Required (Yes/No):** **Description:** Contact Type for Letter generation **Name:** contactClassForLetter **Required (Yes/No):** **Description:** Contact Class for letter generation **Name:** contactMethodForLetter **Required (Yes/No):** **Description:** Contact Method for Letter generation **Name:** contactTypeForSMS **Required (Yes/No):** **Description:** Contact Type for SMS **Name:** contactClassForSMS **Required (Yes/No):** **Description:** Contact Class for SMS **Name:** contactMethodForSMS **Required (Yes/No):** **Description:** Contact Method for SMS
Detailed Design	This algorithm invokes **GenerateContactForPTP** service, which creates the contact (generate Letter or SMS) when PTP moves to **Kept** state.

Table 4–74 PTP Active Algorithm: Sample Algorithm

Algorithm Name	C1-CURENTITY
Parameters	**Name:** contactTypeForLetter **Value:** OVERDUE **Name:** contactClassForLetter **Value:** CCC **Name:** contactMethodForLetter **Value:** OTBL **Name:** contactTypeForSMS **Value:** OVERDUE **Name:** contactClassForSMS **Value:** CCC **Name:** contactMethodForSMS **Value:** OTBS

4.44 PTP Active Algorithm: C1_PTPBRKLS

This section provides details of the PTP Active Algorithm: C1_PTPBRKLS algorithm.

Table 4–75 PTP Active Algorithm: C1_PTPBRKLS

Description	Algorithm to generate letter or SMS on Broken Status
Detailed Description	This algorithm is used to generate letter or SMS when PTP moves to broken state.
Algorithm Entity	PTP Broken Algorithm
Program Type	java

Table 4–75 (Cont.) PTP Active Algorithm: C1_PTPBRKLS

Program Name	com.splwg.ccb.domain.customerinfo.paymentPlan.CollectionPTPBrokenForNgpAlgorithm
Parameters	**Name:** contactTypeForLetter **Required (Yes/No):** **Description:** Contact Type for Letter generation **Name:** contactClassForLetter **Required (Yes/No):** **Description:** Contact Class for letter generation **Name:** contactMethodForLetter **Required (Yes/No):** **Description:** Contact Method for Letter generation **Name:** contactTypeForSMS **Required (Yes/No):** **Description:** Contact Type for SMS **Name:** contactClassForSMS **Required (Yes/No):** **Description:** Contact Class for SMS **Name:** contactMethodForSMS **Required (Yes/No):** **Description:** Contact Method for SMS
Detailed Design	This algorithm invokes GenerateContactForPTP service, which creates the contact (generate Letter or SMS) when PTP moves to Broken state.

Table 4–76 PTP Active Algorithm: Sample Algorithm

Algorithm Name	C1_PTPBRKLS
Parameters	**Name:** contactTypeForLetter **Value:** OVERDUE **Name:** contactClassForLetter **Value:** CCC **Name:** contactMethodForLetter **Value:** OTBL **Name:** contactTypeForSMS **Value:** OVERDUE **Name:** contactClassForSMS **Value:** CCC **Name:** contactMethodForSMS **Value:** OTBS

If you want to generate letter, the following parameters are mandatory:

- contactTypeForLetter
- contactClassForLetter
- contactMethodForLetter

If you want to generate SMS, following parameters are mandatory:

- contactTypeForSMS
- contactClassForSMS
- contactMethodForSMS

If you want to generate both Letter and SMS, following parameters are mandatory:

- contactTypeForLetter
- contactClassForLetter
- contactMethodForLetter
- contactTypeForSMS
- contactClassForSMS
- contactMethodForSMS

4.45 Rule facts populating algorithm: C1-BRLSR

This section provides details of the rule facts populating Algorithm: C1_BRLSR algorithm.

Table 4–77 Rule Facts Populating Algorithm: C1-BRLSR

Description	This algorithm is used to populate the facts required for Rule engine.
Detailed Description	This algorithm populates rule facts for Rule/Ruleset from defined Business Object (BO).
Algorithm Entity	BO Rule Search - Rule Parameter Search
Program Type	java
Program Name	com.splwg.ccb.domain.collection.RuleFactsPopulation
Parameters	**Name:** Input Key1 **Required (Yes/No):** Yes **Description:** Primary Key name of defined BO. **Name:** Input Key2 **Required (Yes/No):** No **Description:** Primary Key name of defined BO. **Name:** Input Key3 **Required (Yes/No):** No **Description:** Primary Key name of defined BO. **Name:** Input Key4 **Required (Yes/No):** No **Description:** Primary Key name of defined BO. **Name:** Input Key5 **Required (Yes/No):** No **Description:** Primary Key name of defined BO.

Table 4–77 (Cont.) Rule Facts Populating Algorithm: C1-BRLSR

Parameters	**Name:** Input B O Name1 **Required (Yes/No):** Yes **Description:** BO name to fetch fact values. If BOName1 is defined then its primary key name must be defined in Input Key 1. Similarly configure other BO names. **Name:** Input B O Name2 **Required (Yes/No):** No **Description:** BO name to fetch fact values. If BOName1 is defined then its primary key name must be defined in Input Key 1. Similarly configure other BO names. **Name:** Input B O Name3 **Required (Yes/No):** No **Description:** BO name to fetch fact values. If BOName1 is defined then its primary key name must be defined in Input Key 1. Similarly configure other BO names. **Name:** Input B O Name4 **Required (Yes/No):** No **Description:** BO name to fetch fact values. If BOName1 is defined then its primary key name must be defined in Input Key 1. Similarly configure other BO names. **Name:** Input B O Name5 **Required (Yes/No):** No **Description:** BO name to fetch fact values. If BOName1 is defined then its primary key name must be defined in Input Key 1. Similarly configure other BO names.
Parameters	**Name:** Bo Fields **Required (Yes/No):** Yes **Description:** Comma separated BO fields of defined BO names. **Name:** Rule Fact Codes **Required (Yes/No):** Yes **Description:** Comma separated fact codes for rule to be executed. BO Fields and Rule Fact codes should be defined in the same order. **Name:** Pre Populated Rule Facts Algorithm Code **Required (Yes/No):** No **Description:** Algorithm code of algorithm holding pre populated facts. Rule facts which cannot be retrieved from BO fields can be pre populated in algorithm. These facts will be appended to input facts for rule under execution. Algorithm type must be defined on algorithm spot 'Rule Execution - Pre Populated Rule Facts' (For more information check sample implementation 'C1-PPSF').
Detailed Design	This algorithm is used to populate rule facts from Business object (BO). Business object fields are fetched using combination of BO name and its respective primary key. Further these values are mapped to rule fact code. Also, pre-populated facts are appended to these values, if provided from external algorithm. These populated facts will act as input to defined rule through soft parameter.

Sample Algorithm

Table 4–78 Sample Algorithm

Algorithm Name	C1-BRLSR
Parameters	**Name:** Input Key1 **Value:** accountId **Name:** Input Key2 **Value:** **Name:** Input Key3 **Value:** **Name:** Input Key4 **Value:** **Name:** Input Key5 **Value:**
	Name: Input B O Name1 **Value:** C1-ACCT-EXTN **Name:** Input B O Name2 **Value:** **Name:** Input B O Name3 **Value:** **Name:** Input B O Name4 **Value:** **Name:** Input B O Name5 **Value:** **Name:** Bo Fields **Value:** productClassCode, overdueAmount **Name:** Rule Fact Codes **Value:** ProductClass, OverdueAmount **Name:** Pre Populated Rule Facts Algorithm Code **Value:**

4.46 Borrower Centric Case Lifecycle

This table provides details of the Borrower Level: C1-ASSODELAC algorithm.

Table 4–79 Borrower Level: C1-ASSODELAC

Description	Associate new delinquent account of the customer
Detailed Description	Associate delinquent accounts where the customer is the main customer to the case.
Algorithm Entity	Case Enter Status
Program Type	java
Program Name	com.splwg.ccb.domain.collection.caseType.specialisedCollections.AssociateDelinque ntAccount
Parameters	
Detailed Design	It is invoked in Pending status of borrower centric case. Transition to Borrower Centricity happens only if a customer has multiple delinquent accounts where he is the main customer only.

This table provides details of the Borrower Level : C1-BRWRSW_Y algorithm.

Table 4–80 Borrower Level : C1-BRWRSW_Y

Algorithm Name	C1-BRWRSW_Y
Parameters	**Name:** Customer Level Switch Name **Value:** BRRWR_SW **Name:** Switch Value **Value:** Y

This table provides details of the Borrower Level : C1-BRWRTRNDF algorithm.

Table 4–81 Borrower Level : C1-BRWRTRNDF

Algorithm Name	C1-BRWRTRNDF
Parameters	**Name:** Wait Days **Value:** 0

This table provides details of the Borrower Level : C1-BRWRSW_N algorithm.

Table 4–82 Borrower Level : C1-BRWRSW_N

Algorithm Name	C1-BRWRSW_N
Parameters	**Name:** Customer Level Switch Name **Value:** BRRWR_SW **Name:** Switch Value **Value:** N

4.47 Update Collection Address on Borrower Panel

This table provides details of the Person Address Update -Pre-Processing: C1-PADDPRE algorithm.

Table 4–83 Person Address Update -Pre-Processing: C1-PADDPRE

Description	Person Address Update - Pre Processing
Detailed Description	This algorithm is hooked in PreprocessBusinessObjectRequestAlgorithmSpot. Business object Name: C1-PERADDRCO. Currently there is no logic inside this algorithm. Implementation team can write their own algorithm in this spot and they can attach this in C1-PERADDRCO
Algorithm Entity	Business Object -Pre-Processing
Program Type	Java
Program Name	com.splwg.ccb.domain.collection.address.PersonCollectionAddressPreProcess
Parameters	
Detailed Design	This algorithm is hooked in PreprocessBusinessObjectRequestAlgorithmSpot. Business object Name: C1-PERADDRCO. Currently there is no logic inside this algorithm. Implementation team can write their own algorithm in this spot and they can attach this in C1-PERADDRCO

This table provides details of the Collection Address Post Processing: C1-PERADDPP algorithm.

Table 4–84 Collection Address Post Processing: C1-PERADDPP

Description	Person Address Update - Post Processing
Detailed Description	This is a reference implementation of Post processing algorithm. Customization team can utilize this hook. This is a sample algorithm without having any logic.
Algorithm Entity	Collection Person Address - Post Process
Program Type	Java
Program Name	com.splwg.ccb.domain.collection.address.CollectionPersonAddressPostProcessing
Parameters	
Detailed Design	This is a reference implementation of Post processing algorithm. Customization team can utilize this hook. This is a sample algorithm without having any logic.

4.48 Update Collection Contact Point

This table provides details of Person Contact Point Update - Pre Processing: C1-PCONTPRE algorithm.

Table 4–85 Person Contact Point Update - Pre Processing: C1-PCONTPRE

Description	Person Contact Point Update - Pre Processing
Detailed Description	Contact Point PreProcessing algorithm is attached on BO pre processing spot. This hook is provided for customization and can be utilized to validate the contact point data.
Algorithm Entity	Business Object - Pre Processing
Program Type	Java

Table 4–85 (Cont.) Person Contact Point Update - Pre Processing: C1-PCONTPRE

Program Name	com.splwg.ccb.domain.collection.address.ContactPreferencePreProcess
Parameters	
Detailed Design	Contact Point PreProcessing algorithm is attached on BO pre processing spot. This hook is provided for customization and can be utilized to validate the contact point data.

This table provides details of Collection Contact Point Update - Post Processing: C1-COLLCONTPOST algorithm.

Table 4–86 Collection Contact POint Update - Post Processing: C1-COLLCONTPOST

Description	Person Contact Point Update - Post Processing
Detailed Description	This is a reference implementation of Post processing algorithm. Customization team can utilize this hook. This is a sample algorithm without having any logic.
Algorithm Entity	Collection Contact Preference - Post Processing
Program Type	Java
Program Name	com.splwg.ccb.domain.collection.address.CollectionContactPointPostProcessingSpot
Parameters	
Detailed Design	This is a reference implementation of Post processing algorithm. Customization team can utilize this hook. This is a sample algorithm without having any logic.

5 Feeder Services

Feeder tables in Collections act as an additional layer to validate incoming data pulled from the host. Since ORMB has its own architecture and framework, incoming data from any host is validated as per ORMB objects standard.

Table 5–1 Feeder Services

Service Name	Method Name	Description	Mandatory Fields
AccountFeederApplicationService	AccountFeederResponse update(SessionContext sessionContext,AccountFeederWrapperDTO accountFeederWrapperDTO) throws FatalException	This service adds or updates account related fields in the feeder table. It handles add, update and delete operations.	hostAcctNumber, srcHostId
AccountHardshipDtlsFeederApplicationService	AccountHardshipDtlsFeederResponse update(SessionContext sessionContext,AccountFeederHardshipDtlsWrapperDTO accountFeederHardshipDtlsWrapperDTO) throws FatalException;	This service adds or updates accounts hardship related fields in the feeder table. It handles add, update and delete operations.	hostAcctNumber, srcHostId, reliefEffDt, reliefExpDt, reliefType, hrshipAppId
AccountArrearFeederApplicationService	AccountArrearFeederResponse update(SessionContext sessionContext,AccountArrearFeederWrapperDTO accountArrearFeederWrapperDTO) throws FatalException;	This service adds or updates account arrears related fields in the feeder table. It handles add, update and delete operations. In case of delete, the service also deletes the record from main table.	hostAcctNumber, srcHostId, referenceVal
AccountWarningIndFeederApplicationService	AccountWarningIndFeederResponse update(SessionContext sessionContext,AccountWarningIndFeederWrapperDTO accountWarningIndFeederWrapperDTO) throws FatalException;	This service adds or updates account warning indicator related fields in the feeder table. It handles add, update and delete operations.	hostAcctNumber, srcHostId
AcctPerFeederApplicationService	AcctPerFeederResponse update(SessionContext sessionContext,AcctPerFeederWrapperDTO acctPerFeederWrapperDTO) throws FatalException;	This service adds or updates account person relationship fields in the feeder table. It handles add, update and delete operations.	hostAcctNumber, srcHostId, hostCustomerNbr

Table 5–1 (Cont.) Feeder Services

Service Name	Method Name	Description	Mandatory Fields
FeederPersonApplicationService	FeederPersonResponse update(SessionContext sessionContext,AccountFeederWrapperDTO accountFeederWrapperDTO) throws FatalException	This service adds or updates party related fields in the feeder table. It handles add, update and delete operations.	srcHostId, hostCustomerNbr
FeederPerAddrApplicationService	FeederPerAddrResponse update(SessionContext sessionContext,FeederPerAddrWrapperDTO) throws FatalException	This service adds or updates party address related fields in the feeder table. It handles add, update and delete operations.	srcHostId, hostCustomerNbr, fdrAddrSeqId, addrTypeCd
FeederPerEmpProfileApplicationService	FeederPerEmpProfileResponse update(SessionContext sessionContext,FeederPerEmpProfileWrapperDTO feederPerEmpProfileWrapperDTO) throws FatalException	This service adds or updates party employment details fields in the feeder table. It handles add, update and delete operations.	srcHostId, hostCustomerNbr, determinantValue, fdrEmpSeqId
FeederContactPrefApplicationService	FeederContactPrefResponse update(SessionContext p_SessionContext, FeederContactPrefWrapperDTO p_FeederContactPrefWrapperDTO) throws FatalException	This service adds or updates party contact preferences fields in the feeder table. It handles add, update and delete operations.	srcHostId, hostCustomerNbr, contactPrefType, contactPointType
FeedePerIdApplicationService	FeedePerIdResponse update(SessionContext p_SessionContext, FeedePerIdWrapperDTO p_FeedePerIdWrapperDTO) throws FatalException	This service adds or updates party ID type related fields, such as driving license and so on in the feeder table. It handles add, update and delete operations.	srcHostId, hostCustomerNbr, idType
GroupFeederApplicationService	GroupFeederResponse update(SessionContext sessionContext,GroupFeederWrapperDTO groupFeederWrapperDTO) throws FatalException	This service adds or updates group related fields in the feeder table. It handles add and update operations.	Group_id, determinantValue,srcHostId
GroupMemberFeederApplicationService	GroupMemberFeederResponse update(SessionContext sessionContext, GroupMemberWrapperDTO groupMemberWrapperDTO) throws FatalException	This service adds or updates group member related fields in the feeder table. It handles add, update and delete operations.	Group_id, srcHostId,determinantValue,Party_id(Host_cust_nbr),party_Name