

National Bank of Greece

Card Management System

Solution Design Batch

# Contents

[**Contents 2**](#_3zhuye3msj3f)

[Introduction 5](#_gjdgxs)

[**Detail Design 6**](#_kzajw46h0t75)

[[UFX\_GL] Clearing process 6](#_3znysh7)

[Functional Description 6](#_2et92p0)

[Way4 7](#_111kx3o)

[Batch Integration Layer [Mainframe z/OS] 8](#_3l18frh)

[[Fee Collection] 33](#_2yr0y9xukqr3)

[Functional Description 33](#_tt8smlodlt2n)

[Way4 33](#_wotxgwa969z)

[Batch Integration Layer [Mainframe z/OS] 34](#_xca4l4gw38hc)

[[Retry of clearing process] 42](#_u1hemnok1tzb)

[Functional Description 42](#_loaycythbd2p)

[Batch Integration Layer [Mainframe z/OS] 42](#_95zbgd3nh8zc)

[[Remittances] 45](#_l5ziz8s1bk8h)

[Functional Description 45](#_vplbygf6bb5n)

[Batch Integration Layer [Mainframe z/OS] 45](#_u45q26d7lhg3)

[[Direct Debit] 48](#_fuu86xcucw86)

[Functional Description 48](#_byuhomrcouzl)

[Batch Integration Layer [Mainframe z/OS] 48](#_ds2kjyam2shy)

[[Massive Loading ETE] 56](#_o7az05s9idqs)

[Functional Description 56](#_c3rw3l61d1q)

[Batch Integration Layer [Mainframe z/OS] 56](#_arvfp7hkvnjm)

[[Massive (Un)Loading Voucher] 65](#_8gx4q5v8xzbe)

[Functional Description 65](#_spbqshptce6t)

[Batch Integration Layer [Mainframe z/OS] 65](#_urveu28iwr30)

[[Massive Loading KA] 76](#_6i7go55avl1d)

[Functional Description 76](#_6vtjreknvcsm)

[Batch Integration Layer [Mainframe z/OS] 76](#_4xtr051zob0g)

[[Credit Loyalty & Go4More] 78](#_9epbryqhvg90)

[Functional Description 78](#_5wghb86izzx1)

[Batch Integration Layer [Mainframe z/OS] 78](#_kwsfz4ea6w35)

[[Massive Issuance KA] 80](#_uj9yhonwxsrj)

[Functional Description 81](#_itmy4jb6cf1k)

[Batch Integration Layer [Mainframe z/OS] 81](#_6itevnncnctw)

[[Way4 Archiving] 86](#_oo6ssg4dnhu)

[Functional Description 86](#_4d417vuqjnjv)

[Batch Integration Layer [Mainframe z/OS] 86](#_3gwkh4qvepab)

[[Recarding Prepaid Visa to MC] 92](#_bu14h18sgqym)

[Functional Description 92](#_e16jjwqbto4q)

[Batch Integration Layer [Mainframe z/OS] 92](#_1p8crr6gbpqq)

[[Merge Sydipel] 95](#_2weubyuv69dc)

[Functional Description 95](#_vwb4v0ha764n)

[Batch Integration Layer [Mainframe z/OS] 95](#_cm44ogkftxdh)

[[Credit Statements] 98](#_nj8js6gixydg)

[Functional Description 98](#_g9d4n4isv592)

[Batch Integration Layer [Mainframe z/OS] 98](#_1sohp1gkyt65)

[[Instant Issuing Debit & Dual cards] 101](#_gsx66ric21ku)

[Functional Description 101](#_j7wppim5wcik)

[Batch Integration Layer [Mainframe z/OS] 101](#_m3jxlnqfuc8l)

[[Massive Issuance Debit cards] 107](#_vduhgxnoe1mq)

[Functional Description 107](#_d8kgnalsakvr)

[Batch Integration Layer [Mainframe z/OS] 107](#_yfp10rbhu54a)

[[Stip Balance] 110](#_2a8fr2ta8wvg)

[Functional Description 110](#_suce3hk5bwjs)

[Batch Integration Layer [Mainframe z/OS] 110](#_d5xgltqafa7t)

[[Embossing Incoming] 114](#_v8wm6fe4uzpk)

[Functional Description 114](#_wp9wla6e58t4)

[Batch Integration Layer [Mainframe z/OS] 114](#_9w7h5psk3cts)

[[Deletion of card] 122](#_xr43bmab394)

[Functional Description 122](#_5nyvewvvpybh)

[Batch Integration Layer [Mainframe z/OS] 122](#_gm0n3f610brf)

[[Massive Issuance/Reissuance Voucher] 125](#_2tbxrvmyxpwn)

[Functional Description 125](#_62lpee89qdqt)

[Batch Integration Layer [Mainframe z/OS] 125](#_es9hkmjk5u2l)

[[Disconnect Account from Debit] 148](#_saywcxupava7)

[Functional Description 149](#_2imgxte7e7qs)

[Batch Integration Layer [Mainframe z/OS] 149](#_adqbgcvgt4ut)

[[Embossing - Debit] 153](#_sk6lgd2p06y3)

[Functional Description 153](#_2jh4o1g8ih6j)

[Batch Integration Layer [Mainframe z/OS] 153](#_he0qwnfoioag)

[[Embossing - Prepaid] 199](#_vkrc6uiutuff)

[Functional Description 199](#_sq0r0ycsp1n)

[Batch Integration Layer [Mainframe z/OS] 199](#_uz70n4u7i2j3)

[[Embossing - Credit] 235](#_c99ek19q4ras)

[Functional Description 235](#_kilb4w8fvk1c)

[Batch Integration Layer [Mainframe z/OS] 235](#_mjpijkilun6n)

[[Embossing - Dual] 266](#_qycmep27wjxg)

[Functional Description 266](#_ldkd4qu2s59b)

[Batch Integration Layer [Mainframe z/OS] 266](#_o7dn0hfd28kd)

[▪ [CRDT.GL\_ACC\_PARM] – Accounting Data Parametrization 297](#_3ep43zb)

[▪ [CRDT.SMARTPIN\_TRACK] – SmartPin Data 297](#_3utrv2usn1hz)

[▪ [KCCT.CRCA\_TPAYMENTS] – Logging payments 298](#_8xfqpdfx6fzv)

[▪ [CRDT.UFX\_DEBLOCK] 299](#_sgn9fb79r3ub)

[▪ [KCCT.CARDS\_PIN] 299](#_h8hygcl559lu)

[▪ [KCCT.CRCA\_TPAYMENTS] 300](#_szvyk2dh12pw)

[▪ [CRDT.PAYMENT\_ORDER] 301](#_hdd8onxe0eln)

[▪ [CRDT.PAYMENT\_ORDER\_HIS] 301](#_qlgyyl9pzlfv)

[▪ [CRDT.PREPAID\_REQ] 302](#_gku5dg361qm7)

[▪ [CRDT.PREPAID\_REQ\_LOA] 303](#_7cp4geyoddft)

[▪ [CRDT.PREPAID\_REQ\_LOA\_HIST] 303](#_tvi3v7z4isa8)

[▪ [CRDT.PREPAID\_REQ\_ISS] 304](#_hz90hdolb23s)

[▪ [CRDT.PREPAID\_REQ\_ISS\_HIST] 305](#_vhe1naegr96p)

[▪ [KCCT.COMMISSIONS\_TRN] 306](#_rzwxz55j9zfa)

[▪ [KCCT.SMARTPIN] 307](#_rmudi71oy2oo)

[▪ [KCCT.SMARTPIN\_HIST] 307](#_hcwbopw5pcz)

[▪ [CRDT.MSG\_UNIQ\_ARC] 308](#_pkyjizkbhtec)

[▪ [CRDT.MSG\_UNIQ\_SESP\_HIST\_ARC] 308](#_a1ik9txu4chw)

[▪ [KCCT.PR\_ADDRS] 309](#_4ldjniyqitcr)

[▪ [CRDT.DEB\_FILE\_EXC] 309](#_tj2zvfpxe4ge)

[▪ [CRDT.MASS\_ISS\_REQ] 309](#_u3e3p2lww38r)

[▪ [KCCT.TRACK\_INFO] 310](#_tv0324240en4)

[▪ [CRDT.LOADING\_MASTER] 311](#_u7hk0o1jonq2)

[▪ [KCCT.FILE\_TRAN] 312](#_57bp8lb2n17d)

[▪ [CRDT.EMBOS\_PARM] 312](#_knr1f464qu7b)

[▪ [CRDT.EMBOS\_COUNTER] 312](#_ij7eqnllmlf1)

[▪ [CRDT.EMBOS\_FILE\_MASTER] 313](#_to0bv07bd3g8)

[▪ [CRDT.EMBOS\_FILE\_DETAIL] 313](#_a6410yz7ji6w)

# 

# Introduction

The purpose of this document is to provide the specifications of the integration layer that will support batch file exchange between various existing NBG systems with the new CMS solution (Way4). It covers batch data flows and required data processing in order to identify batch files exchange, job flows and system responsibilities.

This document has been based on the following project deliverables:

* Analysis of the existing Batch files exchanged between various NBG systems

The document consists of the following main chapters:

In ***Detail Design***, the overall architecture of the Integration Layer is described. This chapter is divided into sub-chapters, one per identified Batch functional area. Each sub-chapter contains the following sections:

* **Functional description:** The overall batch flow is described providing details about the solution
* **Solution design diagram:** Describes the systems involved in the solution and the data flows across systems. Sequential ordering is provided for each data flow. Furthermore, high level functionalities (accounting, reporting etc.) are depicted in the diagram for each system
* **System chapters:** Each chapter regarding systems (WAY4, etc.) contains information about data that must be provided and/or processed by the respective system. Each system has specific responsibilities that are described in a high level approach. These functionalities may involve, financial actions, data synchronization, data transformations, file delivery and reporting. The solution design is organized per system in order to group together all processes and data imports/exports that must be implemented per system. The following system sections are used in the document :
  + **WAY4 (partial parts)**
  + **Batch Integration layer[Mainframe z/OS]**
* **Solution impact**:

In **Appendices**, the following sub-chapters are included :

* **CMS tables**: includes a list of newly introduced data tables which are used to support new functionality. Initial naming, description and structure of these data tables are provided in this section
* **Logging process:** includes a list of transactions for which WAY4 should be updated automatically without an additional interface
* **CMOD reports :** includes a list of all reports that will be produced or decommissioned in the to-be environment

# Detail Design

## [UFX\_GL] Clearing process

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | UFX\_GL |
| **Batch Flow Title** | KRDPD\*32 |
| **Functional Area** | Way4, UFX, GL, OMF Processing |
| **Source Systems** | Way4 |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   Clearing makes the clearing of all transactions on debit cards as well as accounting using deposit services and 2 parametric card tables. |

### 

### 

### Way4

#### [Standard Process] UFX & GL Export Files

| **Base24 process** | |
| --- | --- |
| **Process Name** | Way4 process |
| **Process Action** | * Way4 delivers daily UFX & GL to IL Mainframe (through SSIS packages for transformation of XML to flat files) |
| **Validations** | * N/A |

| **Input Files** | | | | |
| --- | --- | --- | --- | --- |
| **Step** | **Type** | **Target System** | **File Name** | **Description** |
| 1 | Flat | IL Mainframe | CRDP.PD.UFXEX.FTIP.CRD03200.SEQ | UFX File |
| 2 | Flat | IL Mainframe | CRDP.PD.GLEXP.FTIP.CRD03200.SEQ | GL File |

| **Output Files** | | | | |
| --- | --- | --- | --- | --- |
| **Step** | **Type** | **Target System** | **File Name** | **Description** |
| 1 | Flat | IL Mainframe |  | Output file containing records reposted to W4 |
| 2 | Flat | IL Mainframe |  | Output file with exceptions |
| 3 | Flat | IL Mainframe |  | Output file with suspended transactions |

### 

### 

### Batch Integration Layer [Mainframe z/OS]

#### [KRDPD\*32] Clearing Processing

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRDPDX32** | K63CS1 | CRDP.PD.UFXGL.PERP.CRD03200.SEQ | CRDP.PD.CLEAR.SEMT.CRDX3201.SEQ | Retrieve from exceptions file UFX records (type ‘01’) that DO NOT match. |
|  | K63CS2 | CRDP.PD.GLEXP.FTIP.CRD03200.SEQ &  CRDP.PD.CLEAR.SEMT.CRDX3201.SEQ | CRDP.PD.CLEAR.INTT.CRDX3202.SEQ  CRDP.PD.CLEAR.SEMT.CRDX3203.SEQ  CRDP.PD.CLEAR.SEMT.CRDX3204.SEQ | : In this step three file Joins are performed:  1st:common DRNs GL/EXCP  2nd:not matched DRNs GL/EXCP  3rd:not matched DRNs EXCP/GL |
|  | K63CS3 | CRDP.PD.UFXGL.PERP.CRD03200.SEQ  CRDP.PD.CLEAR.INTT.CRDX3202.SEQ | CRDP.PD.CLEAR.INTT.CRDX3205.SEQ  CRDP.PD.CLEAR.SEMT.CRDX3206.SEQ  CRDP.PD.CLEAR.SEMT.CRDX3207.SEQ | Removes from Exception matched UFX  1st:common DRNs between GL/Excp  2nd:Not matched GL/EXPORT  3rd:Not matched EXCP/GL |
|  | K63CS4 | CRDP.PD.CLEAR.SEMT.CRDX3206.SEQ | CRDP.PD.UFXGL.PERP.CRD03200.SEQ | Just copy Input file to Output |
|  | DEALLOC | CRDP.PD.CLEAR.SEMT.CRDX3201.SEQ  CRDP.PD.CLEAR.SEMT.CRDX3203.SEQ  CRDP.PD.CLEAR.SEMT.CRDX3204.SEQ  CRDP.PD.CLEAR.SEMT.CRDX3206.SEQ  CRDP.PD.CLEAR.SEMT.CRDX3207.SEQ  CRDP.PD.CLEAR.INTT.CRDX3205.SEQ | N/A | Delete files using IBM utility IEFBR14 |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRDPD032** | K35CP | CRDP.PD.UFXEX.FTIP.CRD03200.SEQ | N/A | Validates UFX exported file |
|  | K36CP | CRDP.PD.GLEXP.FTIP.CRD03200.SEQ | N/A | Validates GL  exported file |
|  | K61CS1 | CRDP.PD.UFXEX.FTIP.CRD03200.SEQ | CRDP.PD.CLEAR.INTT.CRDX3202.SEQ | Sorts on DRN the UFX file : CRDP.PD.UFXEX.FTIP.CRD03200.SEQ and reformats it to new output file : CRDP.PD.CLEAR.INTT.CRDX3202.SEQ |
|  | K61CS2 | CRDP.PD.GLEXP.FTIP.CRD03200.SEQ | CRDP.PD.CLEAR.SEMT.CRD03202.SEQ | Sorts GL file: CRDP.PD.GLEXP.FTIP.CRD03200.SEQ, reformats it and adds spaces at the end (positions: 195 to 1024) and outputs it to file: CRDP.PD.CLEAR.SEMT.CRD03202.SEQ |
|  | K61CS3 | N/A | CRDP.PD.CLEAR.SEMT.CRD03203.SEQ  CRDP.PD.CLEAR.SEMT.CRD03204.SEQ  CRDP.PD.CLEAR.SEMT.CRD03205.SEQ | 1. Matched records based on their DRN between UFX & GL files are exported to file : CRDP.PD.CLEAR.SEMT.CRD03203.SEQ  2. Not matched UFX to GL records are exported to : CRDP.PD.CLEAR.SEMT.CRD03204.SEQ  3. Not matched GL to UFX records are exported to : CRDP.PD.CLEAR.SEMT.CRD03205.SEQ |
|  | K61CS4 | CRDP.PD.CLEAR.SEMT.CRD03203.SEQ | CRDP.PD.CLEAR.SEMT.CRD03206.SEQ | Sort DRNs and keep unique |
|  | K61CS5 | CRDP.PD.CLEAR.SEMT.CRD03201.SEQ | CRDP.PD.CLEAR.SEMT.CRD03207.SEQ | Find all those DRNs with more than one UFX record |
|  | K61CS6 | CRDP.PD.CLEAR.SEMT.CRD03201.SEQ & CRDP.PD.CLEAR.SEMT.CRD03204.SEQ | CRDP.PD.CLEAR.SEMT.CRD03208.SEQ | Exclude all those UFX records that do not match to their corresponding GL records |
|  | K61CS7 | CRDP.PD.CLEAR.SEMT.CRD03208.SEQ &  CRDP.PD.CLEAR.SEMT.CRD03207.SEQ | CRDP.PD.CLEAR.SEMT.CRD03209.SEQ  CRDP.PD.CLEAR.SEMT.CRD03210.SEQ | In this step the concatenated Input files CRDP.PD.CLEAR.SEMT.CRD03208.SEQ & CRDP.PD.CLEAR.SEMT.CRD03207.SEQ are used to export CRDP.PD.CLEAR.SEMT.CRD03209.SEQ file that holds UFX records (based on DRN) with multiple occurrences and CRDP.PD.CLEAR.SEMT.CRD03210.SEQ with unique occurrences |
|  | K61CS8 | CRDP.PD.CLEAR.SEMT.CRD03210.SEQ | CRDP.PD.CLEAR.SEMT.CRD03211.SEQ | Sort UFX records per DRN & Account of Input file: CRDP.PD.CLEAR.SEMT.CRD03210.SEQ and export to CRDP.PD.CLEAR.SEMT.CRD03211.SEQ |
|  | K61CS9 | CRDP.PD.CLEAR.SEMT.CRD03210.SEQ  CRDP.PD.CLEAR.SEMT.CRD03202.SEQ | CRDP.PD.CLEAR.SEMT.CRD03212.SEQ | Concatenate Input files: CRDP.PD.CLEAR.SEMT.CRD03210.SEQ & CRDP.PD.CLEAR.SEMT.CRD03202.SEQ and export them to CRDP.PD.CLEAR.SEMT.CRD03212.SEQ those GL records that match unique UFX records based on DRN |
|  | K61CS10 | CRDP.PD.CLEAR.SEMT.CRD03204.SEQ    RDP.PD.CLEAR.SEMT.CRD03205.SEQ | CRDP.PD.CLEAR.SEMT.CRD03213.SEQ | Merge UFX & GL records of input CRDP.PD.CLEAR.SEMT.CRD03204.SEQ &  RDP.PD.CLEAR.SEMT.CRD03205.SEQ that DO NOT match and export them to the CRDP.PD.CLEAR.SEMT.CRD03213.SEQ |
|  | K61CS11 | CRDP.PD.CLEAR.SEMT.CRD03213.SEQ | N/A | Count records of Input file: CRDP.PD.CLEAR.SEMT.CRD03213.SEQ and if count more than 70.000 recs, stop processing (unmatched validation) |
|  | K61CS12 | CRDP.PD.CLEAR.SEMT.CRD03211.SEQ | CRDP.PD.CLEAR.SEMT.CRD03214.SEQ  to  CRDP.PD.CLEAR.SEMT.CRD032xx.SEQ | Split file CRDP.PD.CLEAR.SEMT.CRD03211.SEQ to fourteen (14) separate files: CRDP.PD.CLEAR.SEMT.CRD03214.SEQ to CRDP.PD.CLEAR.SEMT.CRD032xx.SEQ having a split limit of 10000 recs |
|  | K61CS13 | CRDP.PD.CLEAR.SEMT.CRD03214.SEQ &  CRDP.PD.CLEAR.SEMT.CRD03212.SEQ | CRDP.PD.CLEAR.SEMT.CRD03224.SEQ | Keep records of GL that are matching to UFX |
|  | K61CS14 | CRDP.PD.CLEAR.SEMT.CRD03215.SEQ  CRDP.PD.CLEAR.SEMT.CRD03212.SEQ | CRDP.PD.CLEAR.SEMT.CRD03225.SEQ | Keep GL records that are Matching to UFX (file B) |
|  | K61CS15 | CRDP.PD.CLEAR.SEMT.CRD03216.SEQ & CRDP.PD.CLEAR.SEMT.CRD03212.SEQ | CRDP.PD.CLEAR.SEMT.CRD03226.SEQ | Keep recs of GL that exist in UFX (file C) |
|  | K61CS16 | CRDP.PD.CLEAR.SEMT.CRD03217.SEQ &  CRDP.PD.CLEAR.SEMT.CRD03212.SEQ | CRDP.PD.CLEAR.SEMT.CRD03227.SEQ | Keep recs of GL that exist in UFX (file D) |
|  | K61CS17 | CRDP.PD.CLEAR.SEMT.CRD03218.SEQ & CRDP.PD.CLEAR.SEMT.CRD03212.SEQ | CRDP.PD.CLEAR.SEMT.CRD03228.SEQ | Keep recs of GL that exist in UFX (file E) |
|  | K61CS17-K61CS27 | Input files in steps K61CS18-K61CS27 | Output files as in steps K61CS18-K61CS27 | Keep recs of GL that exist in UFX (file F) up to O |
|  | K61CS28 | CRDP.PD.CLEAR.SEMT.CRD03214.SEQ | CRDP.PD.CLEAR.INTT.CRD03235.SEQ | Pairs of 14 UFX & GL files merged to 14 Output files |
|  | K61CS29 | CRDP.PD.CLEAR.SEMT.CRD03209.SEQ | CRDP.PD.CLEAR.SEMT.CRD03245.SEQ | Convert trans Type Code to lower case for Reversals |
|  | K61CS30 | CRDP.PD.CLEAR.SEMT.CRD03245.SEQ & CRDP.PD.CLEAR.SEMT.CRD03234.SEQ | CRDP.PD.CLEAR.INTT.CRD03246.SEQ | Merge UFX & GL on DRN/Transtypecode/amount/rectype |
|  | K63CP | CRDP.PD.CLEAR.SEMT.CRD03213.SEQ | CRDP.PD.CLEAR.CMOD.CRD03213.SEQ | Run PLI pgm K63CP to create CMOD for not matching UFX & GL |
|  | CRD55 | CRDP.PD.CLEAR.CMOD.CRD03213.SEQ | N/A | CMOD for not matched UFX/GL |
|  | K63CS1 | CRDP.PD.CLEAR.SEMT.CRD03213.SEQ | CRDP.PD.CLEAR.INTT.CRD03247.SEQ | Copy not matched GL-UFX recs for Exceptions Reconciliation |
|  | K63CS2 | CRDP.PD.UFXGL.PERP.CRD03200.SEQ | CRDP.PD.CLEAR.SEMT.CRD03246.SEQ | Copy PREP file to TEMP |
|  | CLEAR01 | CRDP.PD.UFXGL.PERP.CRD03200.SEQ | N/A | Clear Prep file using PGM CLRSEQ |
|  | K63CS3 | CRDP.PD.CLEAR.SEMT.CRD03213.SEQ & CRDP.PD.CLEAR.SEMT.CRD03246.SEQ | CRDP.PD.UFXGL.PERP.CRD03200.SEQ | Copy Not matched GL/UFX to PREP to get re-processed |
|  | DEALLOC | CRDP.PD.CLEAR.SEMT.CRD032**01**.SEQ up to 34 & 45-51 & 54 up to 57 | N/A | Delete files using IBM utility IEFBR14 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

#### 

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRDPDZ32** | **SORT1** | CRDP.PD.CLEAR.INTT.CRDx3203.SEQ from A to O CONCATENATED | CRDP.PD.CLEAR.SEMT.CRDZ3201.SEQ | Concatenate files and create a single output file |
|  | ALLOC |  | CRDP.PD.CLEAR.SEMT.CRDZ3202.SEQ  CRDP.PD.CLEAR.SEMT.CRDZ3203.SEQ  CRDP.PD.CLEAR.SEMT.CRDZ3205.SEQ | Allocates three files using IBM utility IEFBR14 |
|  | K32CP | CRDP.PD.CLEAR.SEMT.CRDZ3201.SEQ | CRDP.PD.CLEAR.SEMT.CRDZ3202.SEQ  CRDP.PD.CLEAR.SEMT.CRDZ3203.SEQ  CRDP.PD.CLEAR.SEMT.CRDZ3205.SEQ | Reprocess excepted groups |
|  | SORT2 | CRDP.PD.CLEAR.INTT.CRD**A**3202.SEQ from A to O & Z | CRDP.PD.CLEAR.SEMT.CRDZ3204.SEQ | Merge files |
|  | SORT3 | CRDP.PD.CLEAR.SEMT.CRDZ3204.SEQ | CRDP.PD.CLEAR.FTOP.CRDZ3202.SEQ | Populate with Header (FH) & Trailer records(FT) |
|  | BACKUP1 | CRDP.PD.CLEAR.SEMT.CRDZ3203.SEQ | CRDP.PD.CLEAR.GDGT.CRDZ3203.BACK | Backup to Output GDG(+1) |
|  | SORT4 | CRDP.PD.CLEAR.SEMT.CRDZ3203.SEQ | N/A | Abend if it contains > 100.000 recs |
|  | SORT5 | CRDP.PD.CLEAR.INTT.CRD**A**3204.SEQ from A to O & Z | CRDP.PD.CLEAR.SEMT.CRDZ3206.SEQ | Merge files |
|  | K64CP | CRDP.PD.CLEAR.SEMT.CRDZ3203.SEQ | CRDP.PD.CLEAR.CMOD.CRDZ3203.SEQ | Create CMOD file for excepted recs |
|  | CRD56 | CRDP.PD.CLEAR.CMOD.CRDZ3203.SEQ | N/A | Run CMOD for Excepted recs |
|  | K65CP | CRDP.PD.CLEAR.SEMT.CRDZ3206.SE | CRDP.PD.CLEAR.CMOD.CRDZ3207.SEQ | Execute PLI pgm:K65CP to create CMOD with Switched Suspense accounts |
|  | CRD57 | CRDP.PD.CLEAR.CMOD.CRDZ3207.SEQ | N/A | Run CMOD |
|  | K64CS1 | CRDP.PD.CLEAR.SEMT.CRDZ3203.SEQ | CRDP.PD.CLEAR.INTT.CRDZ3203.SEQ | Copy Rejections for Exception reconciliation |
|  | DEALLOC | CRDP.PD.CLEAR.INTT.CRD**A**3202.SEQ from A to K and CRDP.PD.CLEAR.INTT.CRD**A**3203.SEQ from to K & … | N/A | Delete all files using IBM utility IEFBR14 |

| **Output Files** | | | | |
| --- | --- | --- | --- | --- |
| **Step** | **Type** | **Target System** | **File Name** | **Description** |
| 1 | Flat | SSIS | CRDP.PD.CLEAR.FTOP.CRDZ3202.SEQ | Repost |
| 2 | Flat | CMOD | CRDP.PD.CLEAR.CMOD.CRDZ3203.SEQ | CMOD Exception File |
| 3 | Flat | CMOD | CRDP.PD.CLEAR.CMOD.CRDZ3207.SEQ | CMOD Transaction to Suspense accounts |

#### 

#### [**K35CP**] Validation program of UFX input file-refer to JCL KRDPD032, 1st step

#### 

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| K35CP  **Main processing logic explained** | This program reads the UFX exported file and performs the following validations:  · Reads first record of UFX input file and increases the Reading counter    In a loop, processes all records of UFX input file and performs the following tasks:  · If first record (field ROWCODE) is not ‘FH’-header record, program abends, otherwise    · Checks (in procedure PROCESS\_DATA) by invoking Service SCDDFE (selects in table: CRDT.DEB\_FILE\_EXC) if program K35CP exists. If the timestamp of input file is greater than timestamp retrieved from table or both timestamps are on the same date and multiple executions are permitted then update table CRDT.DEB\_FILE\_EXC with the new timestamp by invoking Service UCDDFE otherwise terminate issuing a warning message as to the file was previously processed.    · When (field ROWCODE) is ‘FD’-Detail record update counters    · When (field ROWCODE) is ‘FT’-Trailer record, update counters and retrieve counter in field REST\_DETAILS.    · For any other ROWCODE value except ‘FH’/’FD’/’FT’ issue error and abend with Rollback    · Read Next record. Since the Trailer record has been READ, it will probably raise the End of File condition.    · Close input file    · Perform validations (in procedure PRF\_VAL) after EOF reached:    o Validation IF ^ L\_FOUND\_HEADER **will never be true since it was validated inside the main loop**  o If no trailer present ABEND with appropriate message  o If records counted in the main loop are not equal to those referred in the trailer record, ABEND with appropriate message  o Terminate processing by displaying processing statistics as to the number of records read.    **Recommendations regarding program’s logic**:    · Reading counter RECORDS\_READ1 should be removed from READ\_INPUT\_FILE1 Procedure and be inserted in main loop. This will allow better validations regarding the record types read inside the main loop    · Validation check after EOF “IF ^L\_FOUND\_HEADER” in PRF\_VAL Procedure will never be true since it will be captured in Procedure PROCESS\_DATA |

#### 

#### 

#### [**K36CP**] Validation program of GL input file–refer to JCL KRDPD032, 2nd step

#### 

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| K36CP  **Main processing logic explained** | This program reads the GL exported file and performs the following validations:    · Reads first record of GL input file and increases the Reading counter    In a loop, processes all records of GL input file and performs the following tasks:  · If first record (field RECFIELD.ROWCODE) is not ‘FH’-header record, program abends, otherwise:    · Checks (in procedure PROCESS\_DATA) by invoking Service SCDDFE (selects in table: CRDT.DEB\_FILE\_EXC) if program K36CP exists. **If the timestamp of input file is greater than timestamp retrieved from table or both timestamps are on the same date and multiple executions are permitted (@SCDDFE.ODATA.MULT\_EXEC = 'Y'** ) then **update table CRDT.DEB\_FILE\_EXC with the new timestamp** by invoking Service UCDDFE otherwise terminate issuing a warning message as to the file was previously processed.    · When (field ROWCODE) is ‘FD’-Detail record update counters and sump up amounts for Debit (L\_ROW\_DB) and Credit (L\_ROW\_CR) into a variable (L\_ROWS\_AMNT).    · When (field ROWCODE) is ‘FT’-Trailer record, update counters and retrieve counter of field REST\_DETAILS concerning number of records and field L\_TRL\_SUM with the total amount of all transactions in the file.    · For any other ROWCODE value except ‘FH’/’FD’/’FT’ issue error and abend with Rollback    · Read Next record. Since the Trailer record has been READ, it will probably raise the End of File condition and program will exit its main loop.    · Close input file    · Perform validations (in procedure PRF\_VAL) after EOF reached:    o Validation IF ^ L\_FOUND\_HEADER **will never be true since it was validated inside the main loop**  o If no trailer present ABEND with appropriate message  o If records counted in the main loop are not equal to those referred in the trailer record, ABEND with appropriate message  o If amounts summed up during program processing do not equal to the amount of FT record, issues appropriate message and ABENDS.  o Terminate processing by displaying processing statistics as to the number of records read.    **Recommendations regarding program’s logic**:    · Reading counter RECORDS\_READ1 should be removed from READ\_INPUT\_FILE1 Procedure and be inserted in main loop. This will allow better validations regarding the record types read inside the main loop    · Validation check after EOF “IF ^L\_FOUND\_HEADER” in PRF\_VAL Procedure will never be true since it will be captured in Procedure PROCESS\_DATA |

#### 

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| K63CP  **Main processing logic explained** | This program **reads the Input file : CRDP.PD.CLEAR.SEMT.CRD03213.SEQ with the unmatched UFX & GL records created in JCL KRDPD032 and in step K61CS10, and creates a CMOD Output file with all UFX & GL transactions that DO NOT match with headers and other information**. It runs daily.     * Retrieves the current date and date of the week (using EXEC SQL command) * Writes the headers to the output file * Reads first record of UFX & GL input file and increases the Reading counter * Enters a loop while ^EOF, and in that loop:   + Write the DRN of a UFX or GL record   + Checks for pagination every 25 lines   + Proceeds to Reading the next record * At the end of Input file CLOSES Input & Output files * Prints Statistics as to how many read and written   **This signals the end of processing and control returns to the Operating System** |

#### 

#### [**K61CP**] Updates every group of DRNs transactions with indicator marking the End of Unit of Work

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| K61CP  **Main processing logic explained** | This program **reads the merged Input file : CRDP.PD.CLEAR.INTT.CRD03235.SEQ with the matched UFX & GL records created in JCL KRDPD032 and in step K61CS28, and creates an Output file with the last record of a DRN group marked with an ‘1’ indicator signaling the End of Unit of Work.** More specifically:  Retrieves the current date and date of the week (using EXEC SQL command)   * Opens Input and Output files * Reads first record and performs the following:   + If it is a UFX record type and the previous is not ‘ ‘ it signals a DRN group change (turn L\_GRP\_CHANGE boolean indicator to true)   + Keep current record values to a storage copy   + Update reading counters * Enters a loop while ^EOF, and in that loop:   + If it is a DRN group change :     - Raise indicator to the last record of the Group to ‘1’     - Write to the output file all UFX & GL records of a DRN (array L\_GRP\_ARRAY)     - Keep up counters     - Initialize group of DRNs array   + If it is not a DRN group change then append currently read record to the array checking the array elements NOT to exceed its current size 80 occurrences. * Keep reading till the EOF is signaled * Write last group of DRNs (very important) that is left inside the array while the EOF was raised * Print processing Statistics   **This signals the end of processing and control returns to the Operating System** |
|  |  |

#### 

#### [**K64CP**] Reads the Exemptions file and creates an Output file with Headers and Pagination for CMOD

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| K64CP  **Main processing logic explained** | This program **reads the Input file : CRDP.PD.CLEAR.SEMT.CRD03213.SEQ with the exemption records created by main clearing program K32CP in JCL: KRDPDZ32 and Step: K32CP and creates an Ouput file with Headers and Pagination for CMOD**  The main execution steps of the program are:   * Opens Input and Output files * Retrieves the current date and date of the week (using EXEC SQL command) * Writes the headers to the output file * Reads first record of UFX & GL exempted DRN of the input file and increases the Reading counter * Enters a loop while ^EOF, and in that loop:   + Write to the Output file UFX records ONLY (rec type = ‘01’)   + Checks for pagination every 25 lines   + Proceeds to the Reading of the next record * At the end of Input file CLOSES Input & Output files * Prints Statistics as to how many read and written   **This signals the end of processing and control returns to the Operating System** |

#### 

#### [**K65CP**] Reads the Suspended transactions and creates an Output file with Headers and Pagination for CMOD

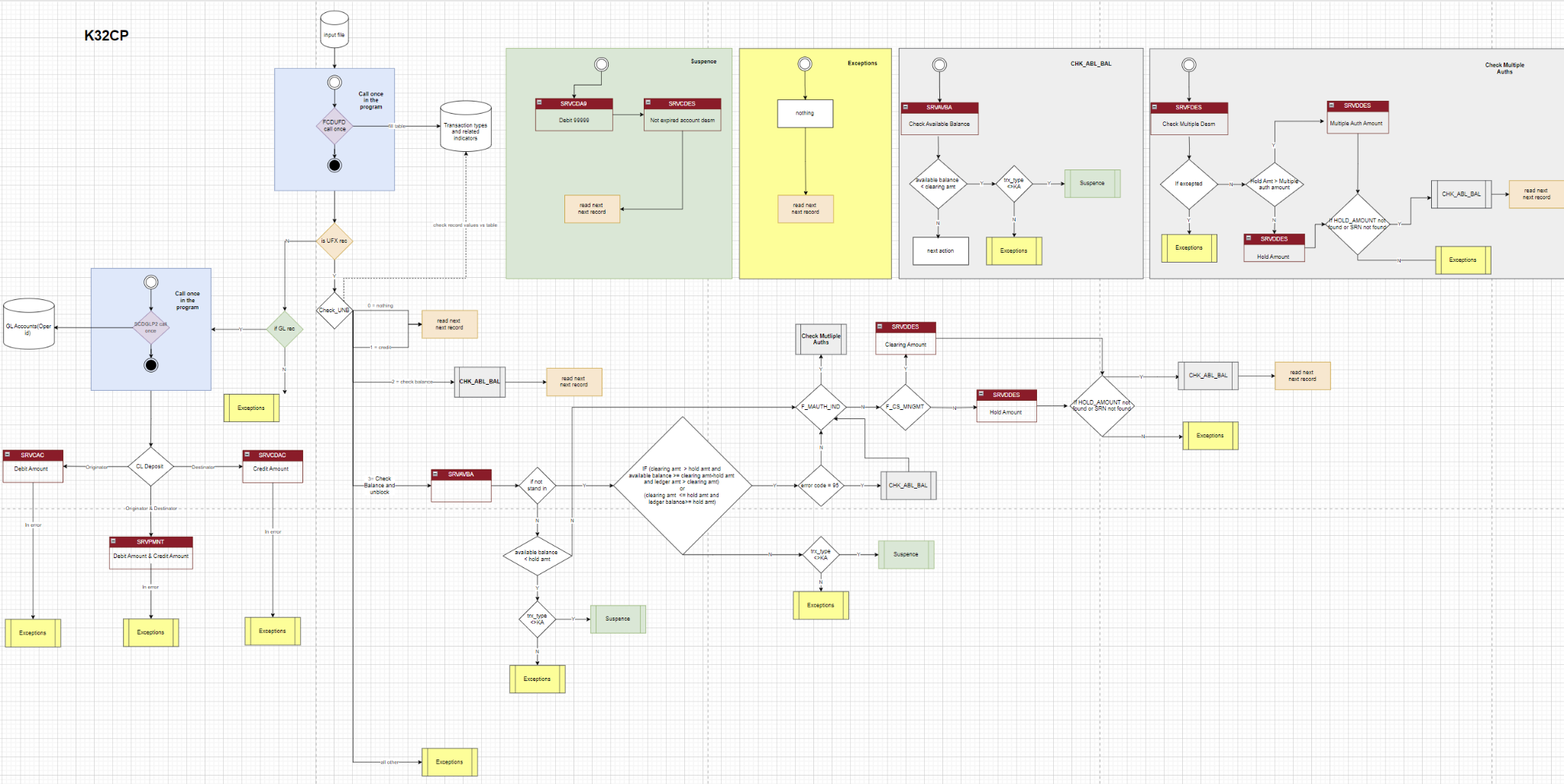
| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| K65CP  **Main processing logic explained** | This program **reads the Input file : CRDP.PD.CLEAR.SEMT.CRDZ3206.SEQ with the Suspended transaction records created by the main clearing program K32CP in JCL: KRDPDZ32 and Step: K32CP and creates an Ouput file with Headers and Pagination for CMOD**  The main execution steps of the program are:   * Opens Input and Output files * Retrieves the current date and date of the week (using EXEC SQL command) * Writes the headers to the output file * Reads first record of UFX & GL exempted DRN of the input file and increases the Reading counter * Enters a loop while NOT EOF, and in that loop:   + Writes to the Output file the record just read   + Checks for pagination every 50 lines   + Proceeds to the Reading of the next record * At the end of Input file CLOSES Input & Output files * Prints Statistics as to how many read and written   **This signals the end of processing and control returns to the Operating System** |

#### 

#### 

#### [K32CP] Clearing program

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPUT FILE DECLARATION** | **INPUT FILE DECLARATION - UFX EXPORT (01) Record Length = 1024**    DCL 1 FINANCIAL\_DOC BASED(ADDR(RECFIELD)),  5 F\_REC\_TYPE CHAR (002) INIT(' '),  5 F\_DRN CHAR (032) INIT(' '),  5 F\_MSG\_CODE CHAR (032) INIT(' '),  5 F\_REQUEST\_CATEGORY CHAR (001) INIT(' '),  5 F\_CHANNEL CHAR (003) INIT(' '),  5 F\_ACCOUNT CHAR (016) INIT(' '),  5 F\_RRN CHAR (032) INIT(' '),  5 F\_SRN CHAR (032) INIT(' '),  5 F\_AUTH\_CODE CHAR (032) INIT(' '),  5 F\_TRANSACTION\_AMOUNT PIC '(13)9V.99' INIT(0),  5 F\_TRANSACTION\_CUR CHAR (003) INIT(' '),  5 F\_BILLING\_AMOUNT PIC '(13)9V.99' INIT(0),  5 F\_BILLING\_AMOUNT\_CUR CHAR (003) INIT(' '),  5 F\_RECONCILIATION\_AMT PIC '(13)9V.99' INIT(0),  5 F\_RECONCILIATION\_AMT\_CUR CHAR (003) INIT(' '),  5 F\_UNBLCK\_AMT PIC '(13)9V.99' INIT(0),  5 F\_UNBLCK\_AMT\_CUR CHAR (003) INIT(' '),  5 F\_TERM\_OWNER\_FIID CHAR (004) INIT(' '),  5 F\_TRANSACTION\_TMSTMP CHAR (019) INIT(' '),  5 F\_TERMINAL\_ID CHAR (016) INIT(' '),  5 F\_ERROR\_CODE CHAR (002) INIT(' '),  5 F\_ORIG\_ACCOUNT\_TYPE CHAR (016) INIT(' '),  5 F\_DEST\_ACCOUNT\_TYPE CHAR (016) INIT(' '),  5 F\_CARD\_NO CHAR (019) INIT(' '),  5 F\_TRX\_TCD CHAR (032) INIT(' '),  5 F\_CS\_MNGMT CHAR (001) INIT(' '),  5 F\_MAUTH\_IND CHAR (001) INIT(' '),  5 F\_MAUTH (15),  7 MSRN CHAR (032) INIT(' '),  5 F\_MERCHANT\_NM CHAR (080) INIT(' '),  5 F\_SOURCE\_PRODUCT CHAR (002) INIT(' '),  5 F\_TARGET\_PRODUCT CHAR (002) INIT(' '),  5 F\_INSTITUTION CHAR (006) INIT(' '),  5 F\_ORIGINATOR\_CHANNEL CHAR (002) INIT(' '),  5 F\_DESTINATION\_CHANNEL CHAR (002) INIT(' '),  5 F\_FILLER CHAR (066) INIT(' ');  **INPUT FILE DECLARATION - GL EXPORT (02)**  DCL 1 GL\_DOC BASED(ADDR(RECFIELD)),  5 G\_REC\_TYPE CHAR (002) INIT(' '),  5 G\_DRN CHAR (032) INIT(' '),  5 G\_SYNTH\_CODE CHAR (032) INIT(' '),  5 G\_CR\_ACCOUNT CHAR (016) INIT(' '),  5 G\_CR\_AMT PIC '(13)9V.99' INIT(0),  5 G\_CRE\_SYNTH\_ACC CHAR (016) INIT(' '),  5 G\_CR\_CURR CHAR (003) INIT(' '),  5 G\_DB\_ACCOUNT CHAR (016) INIT(' '),  5 G\_DB\_AMT PIC '(13)9V.99' INIT(0),  5 G\_DEB\_SYNTH\_ACC CHAR (016) INIT(' '),  5 G\_DB\_CURR CHAR (003) INIT(' '),  5 G\_POST\_DATE CHAR (010) INIT('0001-01-01'),  5 G\_ADATA\_AMT PIC '(13)9V.99' INIT(0),  5 G\_FILLER CHAR (830) INIT(' '); |
| **OUTPUT FILE DECLARATION** | **REPOST FILE** DCL 1 O\_REPSTR,  5 CARD CHAR (019) INIT(' '),  5 DRN CHAR (032) INIT(' ');  **EXCEPTION FILE**  DCL 1 O\_EXCEPT\_REC,  5 RJCT\_FLD CHAR (1024) INIT(' '),  5 ADD\_INFO,  7 GP\_DSC CHAR (003) INIT(' '),  7 GP\_LINE PIC 'Z' INIT(0),  7 SRV\_RC\_PGM CHAR (008) INIT(' '),  7 SRV\_RC CHAR (003) INIT(' '),  7 SRV\_RC\_MSG CHAR (010) INIT(' '),  7 SRV\_RC\_TXT CHAR (080) INIT(' ');  **CMOD INFO 999**  DCL 1 O\_C9FL,  5 DRN CHAR (032) INIT(' '),  5 ACCOUNT CHAR (010) INIT(' '),  5 SRN CHAR (012) INIT(' '),  5 TRANS\_AMNT PIC '(13)9V.99' INIT(0),  5 TRANS\_TMSMP CHAR (019) INIT(' '),  5 TERM\_ID CHAR (016) INIT(' '),  5 CARD\_PAN CHAR (019) INIT(' '),  5 SUSP\_ACC CHAR (010) INIT(' '),  5 TTYCD CHAR (016) INIT(' '),  5 MRCHNT CHAR (080) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | * **L\_ARRAY\_SUSP\_ACC(6)** contains 6 accounts that will be used (depending on the transactions attributes like Channel or Type of Card) for suspended accounts. It is initially filled with values during array declaration. Refer to Appendix-A for specific values/details * **L\_ACT\_ARRAY(11)** it is an “action” array that holds combinations of transaction types and events that are triggered based on them. It is NOT loaded! procedure CALL that will load it, is missing. Refer to Appendix-A for specific values/details * **L\_ACT\_ARRAY\_CHAN(5)** it is an “action” array that holds combinations of transaction channels and events that are triggered based on them (array is improperly positioned-should be re-allocated in the array declaration areas) ). It is loaded at the beginning of program execution. Refer to Appendix-A for specific values/details * **L\_MULTIPLE\_AUTH(16)** array that holds multiple authorization amounts of a specific SRNS. Refer to Appendix-A for specific values/details * **L\_TRC\_ARRAY(500)** holds transaction types along with blocking indicators * **L\_GRP\_ARRAY(80)** used during READ of input file and contains input record values for rectype & drn & uwe of a specific DRN. The array gets initialized every after DRN group is process to hold the next DRN group * **L\_SERV\_ACT(8)** array that holds different combinations of accounts and their usage by a Service Name. It also contains the number of calls (max 2) and the service name for each one. It is a critical array that allows easier program maintenance and operation. Refer to Appendix-A for specific values/details |
| **CRITICAL BOOLEAN INDICATORS** | * **L\_SWITCH\_99** (initialized to False(‘0’B)   **L\_SWITCH\_99**  (balance is insufficient and a *consolidation account* (999) will be used) is set to true when:   * + inside PROC PROCESS\_UFX\_REC when transaction refers to a migrated card   + inside PROC BLS\_CHK (balances checks) when NOT STANDIN & CLEAR\_AMOUNT > HOLD\_AMOUNT & AVAIL\_BALANCE < (CLEAR\_AMT – HOLD\_AMT) & LEDGER\_SUM < CLEAR\_AMT & TRX\_TCD <> ‘KA’   + Inside PROC BLS\_CHK (balances checks) when NOT STANDIN when LEDGER\_SUM < HOLD\_AMOUNT   + Inside PROC BLS\_CHK (balances checks) when NOT STANDIN when   CLEAR\_AMT <= HOLD\_AMT & TRX\_TCD <> ‘KA’   * + Inside PROC BLS\_CHK (balances checks) when STANDIN when AVAIL\_BALANCE < HOLD\_AMT & TRX\_TCD <> ‘KA’   + Inside PROC CHCK\_ABL\_BAL < CLEAR\_AMOUNT & TRX\_TCD <> ‘KA’ * **L\_REPOST** (initialized to False(‘0’B))   **L\_REPOST** is set to true when:   * + inside PROC BLS\_CHK (balances checks) when NOT STANDIN & CLEAR\_AMOUNT > HOLD\_AMOUNT & AVAIL\_BALANCE < (CLEAR\_AMT – HOLD\_AMT) & LEDGER\_SUM < CLEAR\_AMT & TRX\_TCD <> ‘KA’ & L\_FINANCIAL\_DOC.F\_CS\_MNGMT ^= 'Y'   + Inside PROC BLS\_CHK (balances checks) when NOT STANDIN when LEDGER\_SUM < HOLD\_AMOUNT & L\_FINANCIAL\_DOC.F\_CS\_MNGMT ^= 'Y'   + Inside PROC BLS\_CHK (balances checks) when NOT STANDIN when   CLEAR\_AMT <= HOLD\_AMT & TRX\_TCD <> ‘KA’ & L\_FINANCIAL\_DOC.F\_CS\_MNGMT ^= 'Y'   * + Inside PROC BLS\_CHK (balances checks) when NOT STANDIN when   CLEAR\_AMT <= HOLD\_AMT & TRX\_TCD <> ‘KA’ & L\_FINANCIAL\_DOC.F\_CS\_MNGMT ^= 'Y'   * **L\_CHARGE\_ALT9 (**initialized to False(‘0’B))   **L\_CHARGE\_ALT9** is set to true when:   * + Available Balance is less then Billing amount and <> ‘KA’ and case mgmt. = ‘Y’ then indicate the charge of an Alternate 999 Suspense account (otherwise repost transaction to W4) * **L\_MIGRATION (**initialized to False(‘0’B))   **L\_MIGRATION** is set to true when:   * + Inside PROCESS\_UFX\_REC, when F\_CARD\_NO,1,6) = '589242' may be set to true * **L\_GROUP\_CHG (**initialized to False(‘0’B))   **L\_GROUP\_CHG** is set to true when:   * + Inside PROC: READ\_INPUT\_FILE1 , when REC\_TYPE = ‘02’ & RECFIELD.UWE = '1' * **IS\_NOT\_FEE** (initialized to True(‘1’B) inside Boolean function IS\_NOT\_FEE   + IS\_NOT\_FEE **is set to False** when there is an ‘F’ or ‘f’ inside of G\_SYNTH\_CODE * **L\_IS\_DEBIT (**initialized to False(‘0’B))   + **L\_IS\_DEBIT** is set to true when F\_SOURCE\_PRODUCT or F\_TARGET\_PRODUCT = ‘DC’ inside Procedure CHECK\_IF\_DEBIT * **L\_GRP\_EXCEPTED (**initialized to False(‘0’B))   **L\_GRP\_EXCEPTED** is set to true when:   * + In procedure that processes the UFX record when is NOT (Debit movement or Deblock or Balance check)   + When Service requested is not in services array   + When return code from fetching rows (service FCDGLP) of table CRDT.GL\_ACC\_PARM is 20   + When return code from fetching rows (service SCDGLP2) of table CRDT.GL\_ACC\_PARM is 4   + When return code from a Call to Balance of an account (service SRVAVBA) is not 0 and not 8   + When return code from a Call to Clearing Authorization an account (service SRVDDES) is not 0 and not 8 and not (31, 33, 34)   + When return code from a Call to Debit/Credit GL accounts (service SRVLOGI) is not 0 and not 8   + When return code from a Call to Debit/Credit GL accounts of transactions performed in DIAS network (service SRVPMNT) is not 0 and not 8   + When return code from a Call to Debit/Credit Deposit accounts (service SRVLAC5) is not 0 and not 8   + When return code from a Call to Debit/Credit Suspense accounts (service SRVCDA9) is not 0 and not 8   + In all cases that data are written to the Exceptions file   + When return code from a Call to Block a Client Deposit accounts (service SRVCDES) is not 0 and not 8   + In procedure that checks the available balance, if the customers available balance is less than to the Billing amount and is not ‘KA’ (Credit card payment using Debit through ATM) |
| **SERVICES** | * @**SRVDDES** (link area @SRVDDES): Called for amount unblocking * @**SRVFDES** (link area @SRVFDES): Called for retrieving SRN blocked amount * @**SRVAVBA**(link area @SRVAVBA): Called for retrieving of an account Balance * @**SRVPMNT**(link area @SRVPMNT):Called for Debit/Credit Deposit accounts of DIAS transactions * @**SRVLAC5** (link area @SRVLAC5):Called for Debit/Credit Deposit accounts * @**SRVLOGI** (link area @SRVLOGI): Called for Debit/Credit GL accounts * @**SRVCDA9** (link area @SRVCDA9): Called for Debit/Credit Suspence accounts * @**SCDGLP2** (link area @SCDGLP2): Called for Selecting rows from Accounts parameters table CRDT.GL\_ACC\_PARM * **@FCDUFD** fetches rows from table CRDT.UFX\_DEBLOCK regrading DEBLOCKING * **@SCDBRN** (link area @@SCDBRN): Not being used (checks changes from step1 to step2 during migration) * @**SRVCDES** (link area @SRVCDES): Called for Blocking a Deposits Account |
| **Main Process Logic** | Program execution starts with the initialization of the arrays :   * L\_TRC\_ARRAY(500) gets loaded by invoking Service @FCDUFD in INIT\_TRXARY procedure that loads into array fetched rows from CRDT.UFX\_DEBLOCK table, regarding Blocking/Unblocking account      * L\_SERV\_ACT(8) gets loaded in procedure CT INIT\_SERV\_ACT. That table contains the service names that will be invoked to perform actual account movements based on account types namely GL or Deposits. * L\_ACT\_ARRAY\_CHAN(5) gets loaded in procedure INIT\_ACTARR\_CHAN. It contains actions that are performed based on the transaction's channel. * RETR\_CUR\_DT\_TM: Procedure that issues SQL commands for current date and time * Initializes counters (#DBRINIS('K32CP', COUNTERS);) * Opens files   + Input (INPFILE1) file   + Output (OUTFILE1, OUTFILE2, OUTFILE3) * Reads 1st record in PROCEDURE READ\_INPUT\_FILE1 (using # DBRREAD) into record description area RECFIELD (for analytical record structure and all other areas that redefine it, refer to Appendix-B named RECFIELD). The processes inside that procedure are described in detail in the next chapter titled **READ\_INPUT\_FILE1** * When programs reaches the end of Input file :   + Closes all input & Output files   + Issues statistics gathered during processing   + **RETURNS to the Operating System - END**   READ\_INPUT\_FILE1  When :   * + REC\_TYPE = ‘01’ (UFX record) keep DRN in variable L\_GROUP\_DRN and resets Boolean variable L\_GROUP\_CHG to false, increase GROUP\_TOT counter   + REC\_TYPE = ‘02’ increase GL\_TOT, and when UWE = ‘1’ sets Boolean variable L\_GROUP\_CHG to true   Regardless of REC\_TYPE increase counters L\_GP\_ROW and REC\_READ and append values of REC\_TYPE, R\_DRN, FILLER\_1, UWE concatenated into RCRD field of L\_GRP\_ARRAY  After the first record is read in PROC (READ\_INPUT\_FILE1) the program processes sequentially in a loop, the input file till the END. It is important to mention that the Input File contains records in sorted order based on DRN & record type of the transactions of both UFX & GL types. |
| **PROGRAM MAIN LOOP** | If a group is NOT excepted (it is not for the 1st READ), based on the record type (REC\_TYPE) two distinct procedures are executed:  When REC\_TYPE = ‘01’ executes procedure **PROCESS\_UFX\_REC** (procedure logic explained below)  When REC\_TYPE = ‘02’ executes procedure **PROCESS\_GL\_REC\_SETTL**(procedure logic explained below)  When REC\_TYPE <> ‘01’ & ‘02’ process is terminated with ROLLBACK  The two procedures that accomplish the logic processing of the record types ‘01’ & ‘02’ are described in detail  in the following sections.  After processing all the transactions of a group (same DRN) and reaching the end of that group – and before  proceeding to the next READ, the program:  performs a ROLLBACK of the whole group if it has been marked as exempted (variable  L\_GROUP\_CHG is true) during processing of any transaction within the same DRN and reports it to  the EXCEPTIONS file (OUTFILE2)  **OTHERWISE**  If the DRN group of transactions has been marked as 999 (see Boolean variable L\_SWITCH\_99 above) and is not  a migrated card :   * it is written to OUTFILE3 * if it was marked as reposted (see Boolean variable L\_REPOST above), it is exported to OUTFILE1 for a W 4 repost.   At the end of a DRN group counters are updated  In any case the group array (L\_GRP\_ARRAY) gets initialized to hold the next DRN group of transactions.  Finally, the program at this point issues a DBRCKPT and keeps processing (READ) of the next sequential Input  file record.  **PROC: PROCESS\_UFX\_REC (rec\_type = ‘01’) explained**.  The processing of rec\_type = ‘01’ is accomplished in this procedure. The main processing actions that take place (in sequence) are :   * Initialize all critical variables to their data type default values (numeric to zeroes and character to spaces). * Assign values from record area to local variables (L\_FINANCIAL\_DOC = FINANCIAL\_DOC, BY NAME) * Count stats for “No Block-95”, “Standins-98” , “Multiple Pre. Auths” and “Case Mgmt” counters for all records of input file * Determine if Card Holder is in origination or destination or “Not Present” at all   + When F\_ORIG\_ACCOUNT\_TYPE = 'P' then Cardholder (O)   + When F\_DEST\_ACCOUNT\_TYPE = 'D’ then Cardholder (D)   + Otherwise Cardholder = ‘NP’ * Initialize “Transaction Type Code”   + When F\_REQUEST\_CATEGORY = 'R' then L\_TTCD = lower (L\_FINANCIAL\_DOC.F\_TRX\_TCD)   + Otherwise L\_TTCD = L\_FINANCIAL\_DOC.F\_TRX\_TCD * Check if card is a Debit(F\_SOURCE\_PRODUCT = 'DC') and raise the appropriate indicator (L\_IS\_DEBIT) * Determine if transaction unblocking is required when there is (destinator or originator) and is Debit(L\_IS\_DEBIT) by searching Transaction Type Code against all internal table entries L\_TRC\_ARRAY (loaded by invoking Service @FCDUFD-that fetches rows form table CRDT.UFX\_DEBLOCK) in INIT\_TRXARY procedure * Based on channel transaction used and message code determine the action that will be taken (array L\_ACT\_ARRAY\_CHAN and event EVNT) * If bin is 589242 check if it is migrated and if so, switch to 999 suspense account (L\_SWITCH\_99) * If not transaction is NOT Excepted (^L\_GRP\_EXCEPTED) then, Depending on account Unblocking indicator - L\_CHK\_UNB (found from scanning array L\_TRC\_ARRAY that is holding the entries of table CRDT.UFX\_DEBLOCK) when:   + When = ‘0’ for N/A or ‘1’ for “credit Movement” do not take action   + When = ‘2’ **check available balance** only   + When = ‘3’ and is NOT a Reversal (F\_REQUEST\_CATEGORY ^= 'R') and Posted to Client Deposit (F\_DEST\_ACCOUNT\_TYPE,1,1) = ‘P’), it is a **Debit movement** and therefore perform **Unblock as well as Balance check**   + In any other case raise the unrecognizable indicator to the Exceptions file (L\_GRP\_EXCEPTED)   At this point, the UFX record processing gets completed and if the Unit/Group has not been completed, proceed with reading/processing of the next record, otherwise process current unit of work Procedure PROC\_WORK, that will depending on the Indicators either:   * When Excepted (L\_GRP\_EXCEPTED), Roll back Unit of Work (DRN records) and report it to the Exceptions file * If it has been switched to 999-suspense Account(L\_SWITCH\_99) and is not migrated (^L\_MIGRATION) it will be reported to the file CMOD file (OUTFILE3) * If it is about to repost it (L\_REPOST) write to the W4 repost file (OUTFILE1) * In any case take a CHECKPOINT and reset the unit of work array L\_GRP\_ARRAY   Proceed processing by reading the next sequential record of Input file  **PROC: PROCESS\_GL\_REC\_SETTL (rec\_type = ‘02’) explained**.  The processing of rec\_type = ‘02’ that is accomplished in this procedure, happens **when the transaction is not switched to 99 (see Boolean variable L\_SWITCH\_99 above) or is switched to 99 and transaction is not a fee transaction** (Boolean variable is initially set to true and becomes False when G\_SYNTH\_CODE contains ‘F’ or ‘f’). **When these two conditions are met, the following processes take place inside the Procedure**:   * Initialization of variables (Boolean, numeric, character are set to their default data type value – refer to Appendix for the names of the specific variables) * If transaction is not switched to 999 (L\_SWITCH\_99) or is a Debit one (L\_IS\_DEBIT) (F\_SOURCE\_PRODUCT is ‘DC’) then :   + If G\_CRE\_SYNTH\_ACC = 'Cl Deposit' is a Credit Account (L\_GL\_CRE\_ACC\_IS\_CL is set to True) and Type gets ‘DEPO’   + If G\_DEB\_SYNTH\_ACC = 'Cl Deposit' is a Debit Account (GL\_DEB\_ACC\_IS\_CL is set to True) and Type gets ‘DEPO’   + If not a Debit(^L\_GL\_DEB\_ACC\_IS\_CL) or Switched to 999 (L\_SWITCH\_99) fill in values for clearing Debit:     - Prepare values for fetching rows from table CRDT.GL\_ACC\_PARM by invoking Service SCDGLP2 and assign fetched values to their corresponding fields. Remove before the Check Digit of the account     - Repeat the same process as described above for a CREDIT account     - Loop through Services Table Array and retrieve the suitable Service based on the account type determined above (1st dot). Refer to Appendix-A for all combinations of Debit/Credit types that lead to the appropriate Service name.     - Consider a card as Credit(L\_DEB\_CRE\_IND=’C’) , when the account type (L\_GL\_CRE\_ACC\_TYPE) is either ‘DEPO’ or ’LOG1’ otherwise it is a Debit card (L\_DEB\_CRE\_IND=’D’)     - If transaction is NOT excepted(^L\_GRP\_EXCEPTED) then process it based on Credit/Debit Indicator set above (L\_DEB\_CRE\_IND):       * by invoking the FIRST service (L\_SERVICE\_1ST)       * In case of a second Service is declared (L\_NUM\_OF\_ACTIONS > 1), invoke 2nd Service with card type as Credit   At this point, the UFX record processing gets completed and if the Unit/Group has not been completed, proceed with reading/processing of the next record, otherwise process current unit of work Procedure PROC\_WORK, that will depending on the Indicators either:   * When Excepted (L\_GRP\_EXCEPTED), Roll back Unit of Work (DRN records) and report it to the Exceptions file * If it has been switched to 999-suspense Account(L\_SWITCH\_99) and is not migrated (^L\_MIGRATION) it will be reported to the file CMOD file (OUTFILE3) * If it is about to repost it (L\_REPOST) write to the W4 repost file (OUTFILE1) * In any case take a CHECKPOINT and reset the unit of work array L\_GRP\_ARRAY   Proceed processing by reading the next sequential record of Input file |
| **Arrays filled with corresponding values** | ARRAY WITH VALUES FOR SUSPENSE ACCOUNTS  DCL L\_ARRAY\_SUSP\_ACC(6) CHAR (16) INITIAL (  /\* (NBG)ONUS POS \*/ '70499991084',  /\* DIAS POS \*/ '70499991167',  /\* MC ATM/POS \*/ '70499991241',  /\* DIAS ATM \*/ '86999999062',  /\* ALTERNATE 999 \*/ '70499991407',  /\* VISA ATM/POS \*/ '70499991597');  INIT\_SERV\_ACT :PROC TO INITIALIZE ARRAY INCLUDING INFORMATION  REGARDING WHICH SERVICE(S) WILL BE CALLED DEPENDING ON ACCOUNT  TYPE (DEBIT AND CREDIT). FIELD NUM\_OF\_CALLS INDICATES HOW MANY  CALLS WILL BE PERFORMED  *CASE FOR DEBIT DEPOSIT AND CREDIT DEPOSIT*  L\_SERV\_ACT(1).DEB\_ACC\_TYPE = 'DEPO';  L\_SERV\_ACT(1).CRE\_ACC\_TYPE = 'DEPO';  L\_SERV\_ACT(1).NUM\_OF\_CALLS = 1;  L\_SERV\_ACT(1).SERVICE\_1ST = 'SRVPMNT';  L\_SERV\_ACT(1).SERVICE\_2ND = ' ';  *CASE FOR DEBIT DEPOSIT AND CREDIT GENERAL LEDGER ACCOUNT*  L\_SERV\_ACT(2).DEB\_ACC\_TYPE = 'DEPO';  L\_SERV\_ACT(2).CRE\_ACC\_TYPE = 'LOGI';  L\_SERV\_ACT(2).NUM\_OF\_CALLS = 1;  L\_SERV\_ACT(2).SERVICE\_1ST = 'SRVLAC5';  L\_SERV\_ACT(2).SERVICE\_2ND = ' ';  *CASE FOR DEBIT GENERAL LEDGER ACCOUNT AND CREDIT DEPOSIT*  L\_SERV\_ACT(3).DEB\_ACC\_TYPE = 'LOGI';  L\_SERV\_ACT(3).CRE\_ACC\_TYPE = 'DEPO';  L\_SERV\_ACT(3).NUM\_OF\_CALLS = 1;  L\_SERV\_ACT(3).SERVICE\_1ST = 'SRVLAC5';  L\_SERV\_ACT(3).SERVICE\_2ND = ' ';  *CASE FOR DEBIT AND CREDIT GENERAL LEDGER ACCOUNT*  L\_SERV\_ACT(4).DEB\_ACC\_TYPE = 'LOGI';  L\_SERV\_ACT(4).CRE\_ACC\_TYPE = 'LOGI';  L\_SERV\_ACT(4).NUM\_OF\_CALLS = 2;  L\_SERV\_ACT(4).SERVICE\_1ST = 'SRVLOGI';  L\_SERV\_ACT(4).SERVICE\_2ND = 'SRVLOGI';  *CASE FOR DEBIT GENERAL LEDGER ACCOUNT AND DEBIT SUSPENSE*  L\_SERV\_ACT(5).DEB\_ACC\_TYPE = 'LOGI';  L\_SERV\_ACT(5).CRE\_ACC\_TYPE = 'SUSP';  L\_SERV\_ACT(5).NUM\_OF\_CALLS = 2;  L\_SERV\_ACT(5).SERVICE\_1ST = 'SRVLOGI';  L\_SERV\_ACT(5).SERVICE\_2ND = 'SRVCDA9';  *CASE FOR DEBIT SUSPENSE AND CREDIT GENERAL LEDGER ACCOUNT*  L\_SERV\_ACT(6).DEB\_ACC\_TYPE = 'SUSP';  L\_SERV\_ACT(6).CRE\_ACC\_TYPE = 'LOGI';  L\_SERV\_ACT(6).NUM\_OF\_CALLS = 2;  L\_SERV\_ACT(6).SERVICE\_1ST = 'SRVCDA9';  L\_SERV\_ACT(6).SERVICE\_2ND = 'SRVLOGI';  *CASE FOR DEBIT ACCOUNT WITH AUTOMATIC CREDIT CONTRA ACC*  L\_SERV\_ACT(7).DEB\_ACC\_TYPE = 'LOG1';  L\_SERV\_ACT(7).CRE\_ACC\_TYPE = 'LOG2';  L\_SERV\_ACT(7).NUM\_OF\_CALLS = 1;  L\_SERV\_ACT(7).SERVICE\_1ST = 'SRVLOGI';  L\_SERV\_ACT(7).SERVICE\_2ND = ' ';  *CASE FOR CREDIT ACCOUNT WITH AUTOMATIC CREDIT CONTRA ACC*  L\_SERV\_ACT(8).DEB\_ACC\_TYPE = 'LOG2';  L\_SERV\_ACT(8).CRE\_ACC\_TYPE = 'LOG1';  L\_SERV\_ACT(8).NUM\_OF\_CALLS = 1;  L\_SERV\_ACT(8).SERVICE\_1ST = 'SRVLOGI';  L\_SERV\_ACT(8).SERVICE\_2ND = ' ';  INIT\_ACTARR: PROC TO INITIALIZE ACTION ARRAY  *CASE FOR ATMDIA\_1, ATMDIA\_2*  L\_ACT\_ARRAY(1).PTRN = 'DIAS\*';  L\_ACT\_ARRAY(1).EVNT = 'ATMDIA';  *CASE OF NBG ATM*  L\_ACT\_ARRAY(2).PTRN = 'NBOG\*';  L\_ACT\_ARRAY(2).EVNT = 'ATMNBG';  *CASE OF KTHMATOLOGIO*  L\_ACT\_ARRAY(3).PTRN = '\*DpP--2X-FEE-REV';  L\_ACT\_ARRAY(3).EVNT = 'KTHMAT';  *CASE OF INTERNATIONAL ATM/POS*  L\_ACT\_ARRAY(4).PTRN = 'BKNT\*';  L\_ACT\_ARRAY(4).EVNT = 'INTERN';  *CASE OF ON US POS - MELLON*  L\_ACT\_ARRAY(5).PTRN = 'NMO\*';  L\_ACT\_ARRAY(5).EVNT = 'NBGPOS';  *CASE OF ON US POS - CARDLINK*  L\_ACT\_ARRAY(6).PTRN = 'CLNK\*';  L\_ACT\_ARRAY(6).EVNT = 'NBGPOS';  *CASE OF ON US POS - DATACASH*  L\_ACT\_ARRAY(7).PTRN = 'DCAS\*';  L\_ACT\_ARRAY(7).EVNT = 'NBGPOS';  *CASE OF ON US POS - EDPS*  L\_ACT\_ARRAY(8).PTRN = 'EDPS\*';  L\_ACT\_ARRAY(8).EVNT = 'NBGPOS';  *CASE OF ON US POS - 1STDATA*  L\_ACT\_ARRAY(9).PTRN = 'FDDH\*';  L\_ACT\_ARRAY(9).EVNT = 'NBGPOS';  *CASE OF DIAS POS*  L\_ACT\_ARRAY(10).PTRN = 'EFOR\*';  L\_ACT\_ARRAY(10).EVNT = 'DIASPOS';  *CASE OF INTERNATIONAL ATM/POS*  L\_ACT\_ARRAY(11).PTRN = 'VISA\*';  L\_ACT\_ARRAY(11).EVNT = 'INTVIS';  INIT\_ACTARR\_CHAN : PROC TO INITIALIZE ACTION ARRAY BASED ON CHANNEL  *CASE FOR ATMDIA\_1, ATMDIA\_2*  L\_ACT\_ARRAY\_CHAN(1).PTRN = 'Z\*';  L\_ACT\_ARRAY\_CHAN(1).EVNT = 'DIASPOS';  *CASE OF NBG ATM*  L\_ACT\_ARRAY\_CHAN(2).PTRN = '3\*';  L\_ACT\_ARRAY\_CHAN(2).EVNT = 'NBGPOS';  *CASE OF KTHMATOLOGIO*  L\_ACT\_ARRAY\_CHAN(3).PTRN = '\*DpP--2X-FEE-REV';  L\_ACT\_ARRAY\_CHAN(3).EVNT = 'KTHMAT';  *CASE OF INTERNATIONAL ATM/POS*  L\_ACT\_ARRAY\_CHAN(4).PTRN = 'E\*';  L\_ACT\_ARRAY\_CHAN(4).EVNT = 'INTERN';  *CASE OF INTERNATIONAL ATM/POS*  L\_ACT\_ARRAY\_CHAN(5).PTRN = 'V\*';  L\_ACT\_ARRAY\_CHAN(5).EVNT = 'INTVIS'; |
| **CMS TABLES** | CRDT.GL\_ACC\_PARM  CRDT.UFX\_DEBLOCK |



## 

## [Fee Collection]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Fee collection |
| **Batch Flow Title** | KRDPD015 |
| **Functional Area** | Way4, UFX, GL, OMF Processing |
| **Source Systems** | Way4 |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   Fee collection process decides whether a fee for a debit card should be collected or not. It uses client validation services and deposits services for the debit of fee and accounting. |

### Way4

#### [Standard Process] Fee collection Export Files

| **Base24 process** | |
| --- | --- |
| **Process Name** | Way4 process |
| **Process Action** | * Way4 delivers daily Fee collection File to IL Mainframe (through SSIS packages for transformation of XML to flat files) |
| **Validations** | * N/A |

| **Input Files** | | | | |
| --- | --- | --- | --- | --- |
| **Step** | **Type** | **Target System** | **File Name** | **Description** |
| 1 | Flat | IL Mainframe | CRDP.PD.FEECO.FTIP.CRD01501.SEQ | Fee Col |

| **Output Files** | | | | |
| --- | --- | --- | --- | --- |
| **Step** | **Type** | **Target System** | **File Name** | **Description** |
| 1 | Flat | IL Mainframe | CRDP.PD.FEECO.FTOP.CRD01502.SEQ | Output file for Way4 |
| 2 | Flat | IL Mainframe | CRDP.PD.FEECO.FTOP.CRD01504.SEQ | Output file for BAE |

### Batch Integration Layer [Mainframe z/OS]

#### [KRDPD015] Clearing Fee Processing

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRDPD015** | ALLOC |  | CRDP.PD.FEECO.SEMT.CRD01501.SEQ  CRDP.PD.FEECO.SEMT.CRD01502.SEQ  CRDP.PD.FEECO.SEMT.CRD01503.SEQ  CRDP.PD.FEECO.FTOP.CRD01504.SEQ | Allocates 4 files using IBM utility IEFBR14 |
|  | K24CP | DSN=CRDP.PD.FEECO.FTIP.CRD01501.SEQ |  | EXECUTION OF FILE VALIDATION - TOTAL RECS & TOTAL AMOUNTS |
|  | TOOL1 | CRDP.PD.FEECO.FTIP.CRD01501.SEQ | CRDP.PD.FEECO.SEMT.CRD01501.SEQ | CREATE NEW FILE WITH UNIQUE ACCOUNTS FROM FEE COLLECTION FILE |
|  | K19CS1 | CRDP.PD.FEECO.SEMT.CRD01501.SEQ  CRDP.PD.CRTSE.FTIP.CRD02502.SEQ | CRDP.PD.FEECO.SEMT.CRD01502.SEQ | CREATE NEW FILE WITH CARD AND ACCOUNT CONNECTION FROM BANKRUPCY FILE |
|  | K19CS2 | CRDP.PD.FEECO.SEMT.CRD01502.SEQ | CRDP.PD.FEECO.SEMT.CRD01503.SEQ | SORT CREATED FILE BY ACCOUNT ASCENDING |
|  | K19CP | CRDP.PD.FEECO.FTIP.CRD01501.SEQ  CRDP.PD.FEECO.SEMT.CRD01503.SEQ | CRDP.PD.FEECO.FTOP.CRD01502.SEQ  CRDP.PD.FEECO.FTOP.CRD01504.SEQ | EXECUTION OF BATCH K19CP |
|  | DEALLOC | CRDP.PD.FEECO.SEMT.CRD01501.SEQ  CRDP.PD.FEECO.SEMT.CRD01502.SEQ  CRDP.PD.FEECO.SEMT.CRD01503.SEQ |  | DEALLOC OF SEMT FILES using IBM utility IEFBR14 |
|  | REPRO1 | CRDP.PD.FEECO.FTOP.CRD01504.SEQ | EMERP.CRD01504.KRDPD015.D210604.SEQ | BACKUP OUTPUT2 of program K19CP using IBM utility IDCAMS |

#### 

#### [K19CP] Clearing program

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | DCL 1 RECFIELD,  2 ROWCODE CHAR(002) INIT(' '),  2 REST\_DETAILS CHAR(075) INIT(' ');  DCL 1 FH\_INPFILE1 BASED(ADDR(RECFIELD)),  2 FH\_ROWCODE CHAR(002) INIT('FH'),  2 FH\_DATE CHAR(010) INIT(' '),  2 FH\_FILLER CHAR(065) INIT(' ');  DCL 1 FD\_INPFILE1 BASED(ADDR(RECFIELD)),  2 FD\_ROWCODE CHAR(002) INIT('FD'),  2 FD\_FEE\_CODE CHAR(006) INIT(' '),  2 FD\_DRN CHAR(018) INIT(' '),  2 FD\_CONTRACT\_NO CHAR(016) INIT(' '),  2 FD\_RBS\_NO CHAR(016) INIT(' '),  2 FD\_CURRENCY PIC'(03)9' INIT(0),  2 FD\_AMOUNT PIC'(13)9V.99' INIT(0);  DCL 1 FT\_INPFILE1 BASED(ADDR(RECFIELD)),  2 FT\_ROWCODE CHAR(002) INIT('FT'),  2 FT\_TOTAL\_RECS PIC'(07)9' INIT(0),  2 FT\_TOTAL\_AMT PIC'9999999999999.99' INIT(0),  2 FILLER CHAR(052) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | DCL 1 RECFIELD2,  2 REST\_DETAILS CHAR(032) INIT(' ');  DCL 1 FD\_INPFILE2 BASED(ADDR(RECFIELD2)),  2 FD\_CARD\_NUM CHAR(016) INIT(' '),  2 FD\_ACCOUNT\_NO CHAR(016) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | DCL 1 FH\_OUTFILE1,  2 FH\_ROWCODE CHAR(002) INIT('FH'),  2 FH\_ROW\_NUMBER PIC'(07)9' INIT(0),  2 FH\_FILE\_LABEL CHAR(025)  INIT('FEECOLLECTIONIMPORT '),  2 FH\_VERSION CHAR(002) INIT('01'),  2 FH\_FC\_TIMESTAMP,  5 FH\_FC\_TMST\_YYYY CHAR(004) INIT(' '),  5 FH\_FC\_TMST\_FIL1 CHAR(001) INIT('-'),  5 FH\_FC\_TMST\_MM CHAR(002) INIT(' '),  5 FH\_FC\_TMST\_FIL2 CHAR(001) INIT('-'),  5 FH\_FC\_TMST\_DD CHAR(002) INIT(' '),  5 FH\_FC\_TMST\_FIL3 CHAR(001) INIT(' '),  5 FH\_FC\_TMST\_HH CHAR(002) INIT(' '),  5 FH\_FC\_TMST\_FIL4 CHAR(001) INIT(':'),  5 FH\_FC\_TMST\_MI CHAR(002) INIT(' '),  5 FH\_FC\_TMST\_FIL5 CHAR(001) INIT(':'),  5 FH\_FC\_TMST\_SS CHAR(002) INIT(' '),  2 FH\_FILE\_NO PIC'(02)9' INIT(' ');  DCL 1 FD\_OUTFILE1,  2 FD\_ROWCODE CHAR(002) INIT('FD'),  2 FD\_ROW\_NUMBER PIC'(07)9' INIT(0),  2 FD\_DRN CHAR(018) INIT(' '),  2 FD\_CARD\_NO CHAR(016) INIT(' '),  2 FD\_WAIVER\_IND CHAR(001) INIT(' '),  2 FD\_FILLER CHAR(013) INIT(' ');  DCL 1 FT\_OUTFILE1,  2 FT\_ROWCODE CHAR(002) INIT('FT'),  2 FT\_ROW\_NUMBER PIC'(07)9' INIT(0),  2 FT\_RECS\_TOTAL PIC'(07)9' INIT(0),  2 FT\_FILLER CHAR(041) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | DCL 1 FH\_OUTFILE2,  2 FH\_ROWCODE CHAR(002) INIT('FH'),  2 FH\_ROW\_NUMBER PIC'(07)9' INIT(0),  2 FH\_FILE\_LABEL CHAR(025)  INIT('FEECOLLECTIONIMPORT '),  2 FH\_VERSION CHAR(002) INIT('01'),  2 FH\_FC\_TIMESTAMP,  5 FH\_FC\_TMST\_YYYY CHAR(004) INIT(' '),  5 FH\_FC\_TMST\_FIL1 CHAR(001) INIT('-'),  5 FH\_FC\_TMST\_MM CHAR(002) INIT(' '),  5 FH\_FC\_TMST\_FIL2 CHAR(001) INIT('-'),  5 FH\_FC\_TMST\_DD CHAR(002) INIT(' '),  5 FH\_FC\_TMST\_FIL3 CHAR(001) INIT(' '),  5 FH\_FC\_TMST\_HH CHAR(002) INIT(' '),  5 FH\_FC\_TMST\_FIL4 CHAR(001) INIT(':'),  5 FH\_FC\_TMST\_MI CHAR(002) INIT(' '),  5 FH\_FC\_TMST\_FIL5 CHAR(001) INIT(':'),  5 FH\_FC\_TMST\_SS CHAR(002) INIT(' '),  2 FH\_FILE\_NO PIC'(02)9' INIT(' '),  2 FH\_FILLER CHAR(016) INIT(' ');  DCL 1 FD\_OUTFILE2,  2 FD\_ROWCODE CHAR(002) INIT('FD'),  2 FD\_ROW\_NUMBER PIC'(07)9' INIT(0),  2 FD\_DRN CHAR(018) INIT(' '),  2 FD\_CARD\_NO CHAR(016) INIT(' '),  2 FD\_WAIVER\_IND CHAR(001) INIT(' '),  2 FD\_ACTUAL\_AMOUNT PIC'(13)9V.99' INIT(0),  2 FD\_FILLER CHAR(013) INIT(' ');  DCL 1 FT\_OUTFILE2,  2 FT\_ROWCODE CHAR(002) INIT('FT'),  2 FT\_ROW\_NUMBER PIC'(07)9' INIT(0),  2 FT\_RECS\_TOTAL PIC'(07)9' INIT(0),  2 FT\_FILLER CHAR(057) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | @SRVLAC3: Called to DEBIT THE ACCOUNT and Credit the accounting entry  @ SCDGLP: Called to CRDT.GL\_ACC\_PARM  @ SRVSMAS: Called to validate client account  @ SRVIPKP:Called to validate the account product  @ SRVPROD: Called to find the account product  @ SRVIXAL: Called to find the client SYDIPEL  @ SRVIXAF: Called to find all the debit contracts of current SYDIPEL |
| **Main Process Logic** | Program execution starts with the following :   * Get current Date , Time Stamp from the routine GET\_DATETIME * From the above assigns the local variables with the corresponding values   L\_CURRENT\_DATE , L\_CURRENT\_TIME , L\_SIX\_MONTHS\_OLD\_TIMESTAMP,  L\_ONE\_YEAR\_OLD\_TIMESTAMP , Current Date and Time , Six months Timestamp , One Year Timestamp.   * Opens files   + Input (INPFILE1) file   + Output (OUTFILE1, OUTFILE2) * Executes the routines WRITE\_OUT\_FH\_FILE , WRITE\_OUT\_FH\_FILE2 to write the header FILES for OUTFILE1, OUTFILE2 * Read INPFILE1 First Row * Processes sequentially in a loop, the input file till the END – Details of looping in the next paragraph , when fee L\_WAIVER\_FLAG is 'Y' calls the online service to DEBIT THE ACCOUNT with the fees from the routine GET\_OPERATION\_ID (invoce service @SCDGLP which gets OPERATION ID FROM CRDT.GL\_ACC\_PARM FOR SPECIFIC FEE CODE * Executes the routines WRITE\_OUT\_FT\_FILE , WRITE\_OUT\_FT\_FILE2 to write the trailer FILES for OUTFILE1, OUTFILE2 * Closes Files INPFILE1 OUTFILE1 * Releases Db Functions * Executes the routine for the Statistics of the program |
| **PROGRAM MAIN LOOP** | * Process all the input Lines that are not header or Footer (ROWCODE different'FH' & 'FT' ) and initialize WAIVER\_FLAG to Default value 'Y' , assigns FEE\_AMOUNT from input and increase counter RECORDS\_READ (Possible values for L\_WAIVER\_FLAG is 'Y'-fee to be charged , 'N'-no fee , 'F' false status) * If input file raw has *FEE\_CODE 'AF' (FD\_INPFILE1.FD\_FEE\_CODE)* and first 7 characters of card contract CONTRACT\_NO (FD\_INPFILE1.FD\_CONTRACT\_NO) is '5351430' or '5351439' (BUSSINESS CARD) the flow executes procedure LINK\_SRVSMAS wich   + When account not found (21) , account closed (22) assigns L\_WAIVER\_FLAG to 'F'   + Otherwise calls procedure LINK\_SRVPROD and checks account is ETHNIKIS\_OPSEOS (SRVPROD.CODE4 in '3','4','5' ) ,assigns L\_WAIVER\_FLAG to 'N' and increase counter WAIVED\_ETHNIKIS\_OPSEOS otherwise L\_WAIVER\_FLAG remains 'Y'   + If not ETHNIKIS\_OPSEOS and if L\_WAIVER\_FLAG 'Y' call procedure LINK\_SRVIPKP to check Salary Code.   + When *SALARY CODE* (@SRVIPKP.ODATA.ROWS(1).CODE4) in 'RE', 'SE', 'VA', 'GL', 'GV', 'PL', 'PV', 'BE' then L\_WAIVER\_FLAG is set to 'N' and for the case of 'BE' we increase the counter BE\_CNT.   + When *SALARY CODE* (@SRVIPKP.ODATA.ROWS(1).CODE4) is 'BB' then   L\_WAIVER\_FLAG remains 'Y' , increases counter BB\_CNT and checks if first card issued for this account (by calling procedure CHK\_FIRST\_CRD which calls internally procedure GET\_SYDIPEL\_BY\_CARD to give If card is linked with SYDIPEL by giving the L\_SYDIPEL & GET\_ACC\_CARDS procedure to give the Boolean L\_BOOL True if it is the first card and extra call GET\_MIN\_TMSTMP for multiple card to find the first card) , half fee is credited in L\_FEE\_AMOUNT and counter BB\_CNT\_H is increased by 1 else fee is CREDITED FULL AMOUNT and counter BB\_CNT\_F is increased. If no SYDIPEL exists L\_WAIVER\_FLAG is set to 'F'   * + When *SALARY CODE* (@SRVIPKP.ODATA.ROWS(1).CODE4) is 'BP' (same logic as above) then L\_WAIVER\_FLAG remains 'Y' , increases counter BB\_CNT and checks and if first card issued for this account CHK\_FIRST\_CRD (by calling procedure CHK\_FIRST\_CRD which calls internally procedure GET\_SYDIPEL\_BY\_CARD to give If card is linked with SYDIPEL by giving the L\_SYDIPEL & GET\_ACC\_CARDS procedure to give the Boolean L\_BOOL True if it is the first card and extra call GET\_MIN\_TMSTMP for multiple card to find the first card ) , NO fee is credited , L\_WAIVER\_FLAG is set to 'N' and counter BB\_CNT\_ N is increased by 1 else fee is CREDITED FULL AMOUNT and counter BB\_CNT\_F is increased. If no SYDIPEL exists L\_WAIVER\_FLAG is set to 'F'   + For the case that L\_WAIVER\_FLAG not 'F' we write the file OUTFILE1 * If input file raw has *FEE\_CODE* ‘CP’ or ‘IM’ the program calls the procedure for SET\_WAIVER\_FOR\_CP *which sets fee* L\_WAIVER\_FLAG to Y/N according to card class and expired products oldness   + Sets L\_WAIVER\_FLAG to 'N' and call procedure SET\_CUR\_CARD\_CLASS which assigns type of card L\_CUR\_CARD\_CLASS\_DESC to MC DEBIT, VISA DEBIT values 'M' , 'V' correspondingly and after calls GET\_SYDIPEL\_BY\_CARD to give If card is linked with SYDIPEL by giving the L\_SYDIPEL.   + If no SYDIPEL exists L\_WAIVER\_FLAG is set to 'F'   + If SYDIPEL exists, it calls procedure COUNT\_MASTERCARD for Active Debit products (I\_CLSF\_IT\_ACC\_K = '15100005') .     - COUNT\_MASTERCARD : When first 6 Digits are MC Debit ('535142') do :  When the client of the actual card has already other (same scheme) card products, search if the issuing date of cards are same or not. If the issuing day of actual card <> issuing day of other (same scheme) cards then the fee has to be debit L\_WAIVER\_FLAG is set to 'Y'.  If the client attend to issue more than 1 card product at the same day and he has not other card products in his possession, the card product with the last timestamp has to waive L\_WAIVER\_FLAG is set to 'N'. (if he already has other card products then the fee must be debit L\_WAIVER\_FLAG is set to 'Y')   For each MC product the counter MC\_COUNT increases.   * + When not a MC product L\_WAIVER\_FLAG is set to 'F'   + For the case of unique DEBIT MC product or L\_WAIVER\_FLAG is 'N' then we call the procedure to check the oldness of the product CHECK\_EXPIRED\_ PRODUCTS\_OLDNESS from history table based on L\_SYDIPEL     - For the case that we have expired Debit product for the last 6 months we retrieve the product scheme SET\_EXP\_CARD\_CLASS and in a different scheme we set L\_WAIVER\_FLAG to 'N'. If the card has the same scheme we charge the fee and set 'Y' to L\_WAIVER\_FLAG. IF the same card has expired the last 6 months then L\_WAIVER\_FLAG is set to 'N'     - If the client L\_SYDIPEL does not have expired products L\_WAIVER\_FLAG is set to 'N'   + For the case that L\_WAIVER\_FLAG is 'Y' the program Calls procedure LINK\_SRVIPKP as an extra step and repeats the following as below     - When SALARY CODE (@SRVIPKP.ODATA.ROWS(1).CODE4) in 'RE', 'SE', 'VA', 'GL', 'GV', 'PL', 'PV', 'BE' then L\_WAIVER\_FLAG is set to 'N' and for the case of 'BE' we increase the counter BE\_CNT.     - When SALARY CODE (@SRVIPKP.ODATA.ROWS(1).CODE4) is 'BB' then L\_WAIVER\_FLAG remains 'Y' , increases counter BB\_CNT and checks if first card issued for this account (by calling procedure CHK\_FIRST\_CRD which calls internally procedure GET\_SYDIPEL\_BY\_CARD to give If card is linked with SYDIPEL by giving the L\_SYDIPEL & GET\_ACC\_CARDS procedure to give the Boolean L\_BOOL True if it is the first card and extra call GET\_MIN\_TMSTMP for multiple card to find the first card) , half fee is credited in L\_FEE\_AMOUNT and counter BB\_CNT\_H is increased by 1 else fee is CREDITED FULL AMOUNT and counter BB\_CNT\_F is increased. If no SYDIPEL exists L\_WAIVER\_FLAG is set to 'F'     - When SALARY CODE (@SRVIPKP.ODATA.ROWS(1).CODE4) is 'BP' (same logic as above) then L\_WAIVER\_FLAG remains 'Y' , increases counter BB\_CNT and checks and if first card issued for this account CHK\_FIRST\_CRD (by calling procedure CHK\_FIRST\_CRD which calls internally procedure GET\_SYDIPEL\_BY\_CARD to give If card is linked with SYDIPEL by giving the L\_SYDIPEL & GET\_ACC\_CARDS procedure to give the Boolean L\_BOOL True if it is the first card and extra call GET\_MIN\_TMSTMP for multiple card to find the first card ) , NO fee is credited , L\_WAIVER\_FLAG is set to 'N' and counter BB\_CNT\_ N is increased by 1 else fee is CREDITED FULL AMOUNT and counter BB\_CNT\_F is increased. If no SYDIPEL exists L\_WAIVER\_FLAG is set to 'F'   + For the case where L\_WAIVER\_FLAG 'N' we write the file OUTFILE1 , OUTFILE2 |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.GL\_ACC\_PARM |

## [Retry of clearing process]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Retry of clearing process |
| **Batch Flow Title** | KAMPDEBC |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | Mainframe z/OS |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   The retry of clearing process is used to deblock and debit the debit card transactions that did not succeed during the classic clearing process using deposit services for the debit and accounting. |

### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KAMPDEBC] Retry Processing

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KAMPDEBC** | ALLOCAT | N/A | KATP.PD.NDEPO.SEMT.ACCDESM0.SEQ | Allocate using IBM utility IEFBR14 output file |
|  | DELF1 | KATP.PD.ATMDC.FTOP.ACCDESM0.SEQ |  | Initialize file using pgm:CLRSEQ |
|  | SORTS1 | KATP.PD.NDEPO.GDGT.ACCDESM0.BACK(0) | KATP.PD.NDEPO.SEMT.ACCDESM0.SEQ | Create output file by selecting those records that have Suspense codes:48,92,94,94,95,96 and account = ‘704’ and not ‘70401’ |
|  | KF2NP | KATP.PD.NDEPO.SEMT.ACCDESM0.SEQ | KATP.PD.ATMDC.FTOP.ACCDESM0.SEQ  KATP.PD.NDEPO.CMOD.ACCDESM0.SEQ | Processes Suspended transactions |
|  | COPY01 | KATP.PD.NDEPO.CMOD.ACCDESM0.SEQ | KATT.PD.NDEPO.CMOD.ACCDESM0.SEQ | Create a copy using IDCAMS |
|  | CRD59 | KATP.PD.NDEPO.CMOD.ACCDESM0.SEQ |  | Spool file using IRXJCL |
|  | DEALLOC | KATP.PD.NDEPO.SEMT.ACCDESM0.SEQ |  | Deallocate file using IBM utility IEFBR14 |

#### [KF2NP] Retry Program

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | KF2NP1I: Contains suspended transactions.  This INPUT file gets updated with a flag indicating the processing status : 0=successful, 8=Processed but with logical error, ‘ ‘=Not processed)  DCL KF2NP1I FILE RECORD UPDATE;  DCL 1 RKF2NP1I,  5 F\_DESM CHAR(304) INIT(' '),  5 FLG\_UPDATE CHAR(1) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | (KF2NP1O): KATP.PD.ATMDC.FTOP.ACCDESM0.SEQ contains rejected transactions  DCL 1 RKF2NP1O,  3 OAA PIC'(10)9' INIT(0),  3 F01 CHAR(1) INIT(' '),  3 OACCOUNT CHAR(16) INIT(' '),  3 F02 CHAR(1) INIT(' '),  3 OHOLD\_AMOUNT PIC'(13)9V,99' INIT(0),  3 F03 CHAR(1) INIT(' '),  3 OLEDGER\_BAL PIC'(13)9V,99' INIT(0),  3 F04 CHAR(1) INIT(' '),  3 OTRDATE CHAR(10) INIT(' '),  3 F05 CHAR(1) INIT(' '),  3 OHOLD\_REASON CHAR(80) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | DCL OUTFILE1 FILE RECORD OUTPUT; |
| **INTERNAL TABLES/ARRAYS** | Not Used!!! |
| **CRITICAL BOOLEAN INDICATORS** | Not Used!!! |
| **SERVICES** | SRVSMAS: Checks Account properties  SRVAVBA: Checks Account Balance  SRVDDES: Deblocks an Account  SELORCN: Checks Account activity  SRVPMNT: Performs Credit or Debit of an Account  SRVSACC: Returns the Available Accounting Balance |
| **Main Process Logic** |  |
| **PROGRAM MAIN LOOP** | Program KF2NP performs in sequence the following processes   * Opens Input and Output files * Reads the first record and initializes counters * Gets the current Date/Time and the literal of the Date corresponding to it * Writes Headers to OUTFILE1 * In a loop till the end of Input file   + Checks Account properties by invoking Service SRVSMAS and:     - If successful:     - Invokes Service program SRVAVBA to check account balance and **sum up** LEDGER\_BALANCE+CREDIT\_AVAIL\_BAL+OVERDRFT\_LIMIT     - If accounting balance is greater or equal to the commitment amount then deallocate account by invoking Service SRVDDES     - Check operativity of the account by invoking Service SELORCN     - If Account is active and depending on the blocking/holding code use the following Accounts and Operation IDs:      * Populate Service SRVPMNT with all required information and invoke it to perform the actual account movement * If all goes well proceed to SQL Committing work process and update the Input file with flag (FLG\_UPDATE) set to 0. * If any Service Call return code is not 0, meaning an unsuccessful operation:   + updates (rewrites) Input record with RC:8 in FLG\_UPDATE   + Write input record in Rejections file   + Update respective counters based on Service call eg: Insufficient balance, Unable to Unlock account etc. |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A (except those inside the Services) |

## [Remittances]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Remittances |
| **Batch Flow Title** | KRCPD015 |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | DIAS |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   The remittances process is used to credit the prepaid/credit card through DIAS file PACS008, PACS004, as long as provides validations for the client and the card product. |

### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KRCPD015] Remittances Processing

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| KRCPD015 | K50DP | SYNP.PD.DBXCO.INTT.GPP2WAY4.SEQ | CRDP.PD.DIASP.FTOP.CRD05001.SEQ | Remittances processing |
|  | BACK01 | SYNP.PD.DBXCO.INTT.GPP2WAY4.SEQ, | KATP.PD.DIASP.GDGT.CRD05001.BACK(+1) | CREATE OAFP FILE BACKUP |
|  | BACK02 | CRDP.PD.DIASP.FTOP.CRD05001.SEQ | KATP.PD.DIASP.GDGT.CRD05002.BACK(+1) | CREATE CRDP FILE BACKUP |

#### [K50DP] Remittances Program

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILEFILE DECLARATION** | DCL 1 HRK50DPI BASED(ADDR(RK50DPI)),  2 CARD\_PAN CHAR(19) INIT(' '),  2 AMOUNT\_CURR CHAR(03) INIT(' '),  2 TRANS\_AMOUNT PIC '(13)9V.99' INIT(0),  2 VALUE\_DATE CHAR(10) INIT(' '),  2 CREDITOR\_NAME CHAR(30) INIT(' '),  2 DEBITOR\_NAME CHAR(30) INIT(' '),  2 ACQ\_BANK CHAR(11) INIT(' '),  2 UNIQUE\_ID\_PACS8 CHAR(35) INIT(' '),  2 DEBIT\_ACCOUNT CHAR(35) INIT(' '),  2 REMITTANCE\_INFO CHAR(140) INIT(' '),  2 PRODUCT CHAR(20) INIT(' '),  2 REFERENCE\_ID CHAR(20) INIT(' '),  2 OPERATION\_ID CHAR(05) INIT(' '),  2 RETURN\_CODE CHAR(02) INIT(' '),  2 RETURN\_TXT CHAR(30) INIT(' '); |
| **OUTFILEFILE DECLARATION** | DCL 1 HRK50DPO\_HEADER BASED(ADDR(RK50DPO)),  2 HROW\_CODE CHAR(02) INIT('FH'),  2 HDESCR CHAR(25) INIT('CARDS DIAS EMBASMATA '),  2 HDATE\_EXEC CHAR(10) INIT('0001-01-01'),  2 FILLER02 CHAR(01) INIT(' '),  2 HTIME\_EXEC CHAR(08) INIT('00.00.00'),  2 FILLER03 CHAR(121) INIT(' ');    DCL 1 HRK50DPO BASED(ADDR(RK50DPO)), /\* RL=167 \*/  2 TROW\_CODE CHAR(02) INIT('FD'),  2 MSGCODE CHAR(15) INIT(' '),  2 DESCRIPTION CHAR(80) INIT(' '),  2 CONTRACTTYPE CHAR(20) INIT(' '),  2 CARD CHAR(19) INIT(' '),  2 AMOUNT\_CURR CHAR(03) INIT(' '),  2 TRANS\_AMOUNT PIC '(13)9V.99' INIT(0),  2 RRN CHAR(12) INIT(' ');    DCL 1 HRK50DPO\_TRAILER BASED(ADDR(RK50DPO)),  2 TROW\_CODE CHAR(02) INIT('FT'),  2 TRNS\_COUNTER PIC '(09)9' INIT(0),  2 TRNS\_SUM\_AMT PIC '(13)9V.99' INIT(0),  2 FILLER CHAR(140) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | @SRV44NP (link area @ SRV44NP): Called to Define MIGRATED & NON-MIGRATED CARDS    @KCTINS (link area @ KCTINS): Called to insert Data INSERT INTO KCCT.CRCA\_TPAYMENTS    @SELTMST(link area @ SELTMST):Calculates THE CURRENT TIMESTAMP    @SRVIXAS(link area @ SRVIXAS): Called to find the client SYDIPEL Relationship with the card    @SCARD1(link area @SCARD1):: Calls the old process of EMBASMATA    @SRVLAC3(link area @ SRVLAC3): Logistics ROUTINE    @SRVONBL(link area @ SRVONBL): onboarding compliance issues |
| **Main Process Logic** | Program execution starts with the following :   * Opens Input , Output Files K50DP1 , K50DPO * Get the current timestamp from the routine SEL\_TIMESTAMP * Write the header to the Output File * Read File * Execute the loop until end of file as described below * Write the trailer to the Output File * Closes Input , Output Files K50DP1 , K50DPO * Writes statistics of counters |
| **PROGRAM MAIN LOOP** | * Process all the input Lines record by record * For each record, it checks if the RETURN\_CODE is blank, indicating a valid transaction otherwise it read next line of the input file * Increase counter WS\_READ * It processes from the input record the CARD\_PAN using the PROC\_SRV44NP subroutine which calls the service @SRV44NP , which determines if the card is SESPIKA or not.@SRV44NP is a service based on @SRVIXAL , @SCCCPN2 pin existence with ID , @SCCCPN pin existence with card and defines the IN\_SESPIKA * flag. * It compares the first six digits of CARD\_PAN are ‘442317' and rewrite the input file with RETURN\_CODE = '99' , RETURN\_TXT = 'ERROR CARD' * Otherwise Depending of the service response @SRV44NP output field IN\_SESPIKA as described above when 'Y' calls the subroutine PROC\_SCARD1 , when 'N' calls PROC\_W4\_WAY and when 'E' rewrite the input file with RETURN\_CODE = '99' , RETURN\_TXT = 'ERROR CARD'. * subroutine PROC\_SCARD1 :For the case that card in SESPIKA calls the service @SCARD1 for old process of Remittance - EMBASMATA but this will never happen so it can be omitted as step * Subroutine PROC\_W4\_WAY : For the case of migrated card to W4 PROC\_W4\_WAY routine first assigns RETURN\_CODE = '00' to input file and execute the services   + @SRVIXAS for SYDIPEL existence with CARD\_PAN as input   + @KCTINS that creates a Transaction KEY having input reference number REFNO the PRODUCT+CARD\_PAN+ VALUE\_DATE and the T\_ACCOUNT input depending on the PRODUCT (when 'PACS008' value '83977' , when 'PACS004\_BAL' then '83667' , when 'PACS004\_BILL' then '83974')   + @SRVLAC3 for Accounting LOGISTIKI in W4 with specific input ,   + @SRVONBL for Checking Onboard Limit. Calling the service @SRVINDS determines if @SRVINDS.I\_CLSF\_INDC\_VAL\_K = 8300001 then ONBOARD\_FLAG = 'Y' and calls the @SRVWLIM to set it.   + Writes the Data for output file for W4 * Rewrites input File with RETURN\_CODE , RETURN\_TXT   Increase counter WS\_REWRITE |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | KCCT.CARDS\_PIN  KCCT.CRCA\_TPAYMENTS |

## [Direct Debit]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Direct Debit |
| **Batch Flow Title** | KRCPD020 - KRCPD021 |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | Way4 |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   The direct debit process is used to debit the Client deposit account, which is connected to his credit card, when the client chooses to be billed monthly via standing order. |

### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KRCPD020] Direct Debit Request Processing

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| KRCPD020 | K40DP | CRDP.PD.CCDDE.FTIP.CRD02001.SEQ | N/A | This step validates the Input file exported from Way4 with Direct Debits transactions of Credit Cards. Standard validations are applied-refer to program down below for details |
|  | K41DP | CRDP.PD.CCDDE.FTIP.CRD02001.SEQ | Table:INSERT TO CRDT.PAYMENT\_ORDER | Insert into DB2 table the transactions of the Input file |

#### [K40DP] Direct Debit Program (Input file Validations)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | **FILEIN**  DCL FILEIN\_LINE CHAR(163);  DCL 1 REC\_HEADER BASED(ADDR(FILEIN\_LINE)),  2 REC\_TYPE CHAR(02),  2 FILE\_TMSTMP CHAR(26),  2 FILLER1 CHAR(135);  /\* INPUT FILE DECLARATION - DIERCT DEBIT FILE \*/  DCL 1 REC\_DETAILS BASED(ADDR(FILEIN\_LINE)),  2 REC\_TYPE CHAR (002),  2 SERVICE CHAR (008),  2 OPERATION\_ID CHAR (005),  2 DEB\_ACC\_IND CHAR (001),  2 REF\_ID CHAR (032),  2 SRN CHAR (032),  2 ISSUING\_CONTRACT CHAR (032),  2 ACCOUNT\_DEB CHAR (016),  2 POSTING\_DATE CHAR (010),  2 AMOUNT PIC '(13)9V.99',  2 BATCH\_G\_SPECIAL CHAR (003),  2 TRT\_CDE CHAR (003),  2 FLAG CHAR (001),  2 FILLER CHAR (002);  DCL 1 REC\_TRAILER BASED(ADDR(FILEIN\_LINE)),  2 REC\_TYPE CHAR(02),  2 TRN\_COUNT CHAR(07),  2 TRN\_AMOUNT PIC '(13)9V.99',  2 FILLER1 CHAR(138); |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | EOF : Raised when end of file is reached |
| **SERVICES** | SCDDFE: Select from CRDT.DEB\_FILE\_EXC  UCDDFE: Updates CRDT.DEB\_FILE\_EXC |
| **Main Process Logic** | This is a standard validation program that checks Input file for the existence of Header, Details and Footer records. It also checks if it can be run on the same date multiple times. Finally, checks number of detail recs and total sums compared to those referred to the trailer record. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place (Header record already checked for existence and multi-run on the same date indicator):   * When record type is ‘FD’ increase counter for records read sum sum record amount * When record type is ‘FT’ check number of records processed and sum of amounts against those specified in ‘FT’ record. If they match, terminate normally, otherwise issue error and stop processing. |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.DEB\_FILE\_EXC gets updated with timestamp of run |

#### [K41DP] Direct Debit Program

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | **FILEIN**  DCL FILEIN\_LINE CHAR(163);  DCL 1 REC\_DETAILS BASED(ADDR(FILEIN\_LINE)),  2 REC\_TYPE CHAR (002),  2 SERVICE CHAR (008),  2 OPERATION\_ID CHAR (005),  2 DEB\_ACC\_IND CHAR (001),  2 REF\_ID CHAR (032),  2 SRN CHAR (032),  2 ISSUING\_CONTRACT CHAR (032),  2 ACCOUNT\_DEB CHAR (016),  2 POSTING\_DATE CHAR (010),  2 AMOUNT PIC '(13)9V.99',  2 BATCH\_G\_SPECIAL CHAR (003),  2 TRT\_CDE CHAR (003),  2 FLAG CHAR (001),  2 FILLER CHAR (002); |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | EOF : Raised when end of file is reached |
| **SERVICES** | ICDPOR: Inserts into table: CRDT.PAYMENT\_ORDERS payment order |
| **Main Process Logic** | Reads input file and when reaches the last record (rec. Type ‘FT’) populates ICDPOR data area and invokes it, to INSERT into DB2 table the ‘FT’ data of the file. |
| **PROGRAM MAIN LOOP** | Reads input file and when reaches the last record (rec. Type ‘FT’) populates ICDPOR data area and invokes it, to INSERT into DB2 table the ‘FT’ data of the file. |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.PAYMENT\_ORDERS |

#### [KRCPD021] Direct Debit Response Processing

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRCPD021** | K42DP |  |  | Execute Direct Debit request for Consumers |
|  | K72DP |  |  | Execute Direct Debit request for Business |
|  | ALLOC1 | N/A | CRDP.PD.CCDDI.FTOP.CRD02101.SEQ | Allocate file using IBM utility IEFBR14 |
|  | K43DP | DB2 tables | CRDP.PD.CCDDI.FTOP.CRD02101.SEQ | Write to output files successful payments of the above processing |
|  | ALLOC1 |  | CRDP.PD.CCDDE.FTOP.CRD02101.SEQ | Allocate file using IBM utility IEFBR14 |
|  | K73DP | DB2 tables | CRDP.PD.CCDDE.FTOP.CRD02101.SEQ | Write to output files unsuccessful payments of the above processing |
|  | K44DP | CRDT\_PAYMENT\_ORDER | CRDT.PAYMENT\_ORDER\_HIS | Move rows to History table |

#### [K42DP] Direct Debit Program (for CONSUMERS)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | SELECT  FILE\_ID, RECORD\_ID,ERROR\_MSG, SRN, ACCOUNT, DEP\_ACCOUNT,  PAYMENT\_AMOUNT,PAYED\_AMOUNT, PAYMENT\_METHOD,  DUE\_DATE, PAYMENT\_STATUS,  UPDATE\_TIMESTAMP, INSERT\_TIMESTAMP  FROM CRDT.PAYMENT\_ORDER  WHERE DUE\_DATE=CURRENT\_DATE AND  PAYMENT\_STATUS='0' AND  PRODUCT\_CODE ='CONSUMER'; |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | SRVAVBA: Retrieve an Account Balance  SRVLAC3: Debit or Credit a Deposits Account |
| **Main Process Logic** | Declares a DB2 Cursor for Consumer Direct Debit Payments for table CRDT.PAYMENT\_ORDER and processes it. Opens and fetches the first row of the cursor. |
| **PROGRAM MAIN LOOP** | The main steps performed are:   * Issues a SAVEPOINT for every fetched row * Retrieves the Account’s Balance * If balance is less than the Due Amount consider as paid amount the available otherwise use the due amount of the transaction * Assign Fetched value to SRVLAC3 data area and Call Service SRVLAC3 to perform the actual Debit of the CL account and Credit the GL account or Vice Versa * If SRVLAC3 was **successful** and the update of table CRDT.PAYMENT\_ORDER with the Payment Amount and the PAYMENT\_STATUS = ‘1’ was **successful** COMMIT work * If SRVLAC3 was **successful** and the update of table CRDT.PAYMENT\_ORDER with the Payment Amount and PAYMENT\_STATUS = ‘1’ was **unsuccessful**  ROLLBACK work * If SRVLAC3 was **unsuccessful** then ROLLBACK work and update of table CRDT.PAYMENT\_ORDER with PAYMENT\_STATUS = ‘3’ and COMMIT work * If SRVAVBA was unsuccessful then update of table CRDT.PAYMENT\_ORDER with PAYMENT\_STATUS = ‘3’ and COMMIT work * At the end of a fetched row processing, Release SAVEPOINT for the next row * Repeat loop steps for the next row UNTIL SQLCODE <> 0. When processing of the cursor reaches its end an SQLCODE=100 will be returned and processing will finish * At end (SQLCODE=100), Closes the Cursor and prints the Statistics |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.PAYMENT\_ORDER |

#### 

#### [K72DP] Direct Debit Program (for BUSINESSES)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | SELECT  DEP\_ACCOUNT AS DEP\_ACCOUNT,  SUM(ABS(PAYMENT\_AMOUNT)) AS PAYMENT\_AMOUNT,  MIN(SRN) AS SRN,  COUNT(\*) AS ICOUNTER  FROM CRDT.PAYMENT\_ORDER  WHERE DUE\_DATE =CURRENT\_DATE AND  PAYMENT\_STATUS='0' AND  PRODUCT\_CODE ='BUSINESS' |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | SRVAVBA: Retrieve an Account Balance  SRVLAC3: Debit or Credit a Deposits Account |
| **Main Process Logic** | Declares a DB2 Cursor for Businesses Direct Debit Payments for table CRDT.PAYMENT\_ORDER and processes it. Opens and fetches the first row of the cursor. |
| **PROGRAM MAIN LOOP** | The main steps performed are:   * Issues a SAVEPOINT for every fetched row * Retrieves the Account’s Balance * If balance is less than the Due Amount consider as paid amount the available otherwise use the due amount of the transaction * Assign Fetched value to SRVLAC3 data area and Call Service SRVLAC3 to perform the actual Debit of the CL account and Credit the GL account or Vice Versa * If SRVLAC3 was **successful** and the update of table CRDT.PAYMENT\_ORDER with the Payment Amount and the PAYMENT\_STATUS = ‘1’ was **successful** COMMIT work * If SRVLAC3 was **successful** and the update of table CRDT.PAYMENT\_ORDER with the Payment Amount and PAYMENT\_STATUS = ‘1’ was **unsuccessful**  ROLLBACK work * If SRVLAC3 was **unsuccessful** then ROLLBACK work and update of table CRDT.PAYMENT\_ORDER with PAYMENT\_STATUS = ‘3’ and COMMIT work * If SRVAVBA was unsuccessful then update of table CRDT.PAYMENT\_ORDER with PAYMENT\_STATUS = ‘3’ and COMMIT work * At the end of a fetched row processing, Release SAVEPOINT for the next row * Repeat loop steps for the next row UNTIL SQLCODE <> 0. When processing of the cursor reaches its end an SQLCODE=100 will be returned and processing will finish * At end (SQLCODE=100), Closes the Cursor and prints the Statistics |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.PAYMENT\_ORDER |

#### [K43DP] Direct Debit Program (Extract Successful Direct Debit transactions)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | SELECT  FILE\_ID, RECORD\_ID,SRN,ACCOUNT, DEP\_ACCOUNT,  PAYMENT\_AMOUNT,PAYED\_AMOUNT, PAYMENT\_METHOD,  DUE\_DATE, PAYMENT\_STATUS,  UPDATE\_TIMESTAMP, INSERT\_TIMESTAMP  FROM CRDT.PAYMENT\_ORDER  WHERE DUE\_DATE=CURRENT\_DATE AND  ( PAYMENT\_STATUS='1' OR PAYMENT\_STATUS='2'); |
| **OUTFILE1 FILE DECLARATION** | **FILEOUT**  DCL LINE CHAR(165) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve (via a DB2 Cursor) all Successful Direct Debit transactions (STATUS = ‘1’ or ‘2’) to an Output file for the Current Date.. |
| **PROGRAM MAIN LOOP** | The main steps performed are:   * Open Cursor against table CRDT.PAYMENT\_ORDER for transactions of the Current Date and with successful payment status ‘1’ or ‘2’ * Write a Header with Current Date and Time * Fetch first row and in a loop till the end of Cursor   + Write Detail row   + Fetch next row * When out of Cursor with SQLCODE = 100   + Write a Trailer record with the number of Cursor Read   + Close Output file   + Close Cursor   + End of Program |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.PAYMENT\_ORDER |

#### [K73DP] Direct Debit Program

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | SELECT  FILE\_ID, RECORD\_ID,ERROR\_MSG,SRN,ACCOUNT, DEP\_ACCOUNT,  PAYMENT\_AMOUNT,PAYED\_AMOUNT, PAYMENT\_METHOD,  DUE\_DATE, PAYMENT\_STATUS,  UPDATE\_TIMESTAMP, INSERT\_TIMESTAMP  FROM CRDT.PAYMENT\_ORDER  WHERE DUE\_DATE=CURRENT\_DATE AND  ( PAYMENT\_STATUS='3'); |
| **OUTFILE1 FILE DECLARATION** | **FILEOUT**  DCL LINE CHAR(300) INIT(' ');  DCL 1 REC\_DETAILS BASED(ADDR(LINE)),  2 REC\_TYPE CHAR (002),  2 RECORD\_ID CHAR (032),  2 ERROR\_MSG CHAR (100),  2 SRN CHAR (032),  2 DEP\_ACCOUNT CHAR (016),  2 PAYED\_AMOUNT PIC '(13)9V.99',  2 DUE\_DATE CHAR (010),  2 ISSUING\_CONTRACT CHAR (032),  2 PAYMENT\_AMOUNT PIC '(13)9V.99',  2 FILLER CHAR (044); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieves all unsuccessful Credit Direct Debit transactions STATUS = ‘3’ and writes them to the output file to be sent back to WAY4 |
| **PROGRAM MAIN LOOP** | The main steps performed are:   * Open Cursor against table CRDT.PAYMENT\_ORDER for transactions of the Current Date and with successful payment status ‘3’ * Write a Header with Current Date and Time * Fetch first row and in a loop till the end of Cursor   + Write Detail row   + Fetch next row * When out of Cursor with SQLCODE = 100   + Write a Trailer record with the number of Cursor Read   + Close Output file   + Close Cursor   + End of Program |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.PAYMENT\_ORDER |

#### [K44DP] Direct Debit Program

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | SELECT  FILE\_ID, RECORD\_ID,SRN,ERROR\_MSG, ACCOUNT, DEP\_ACCOUNT,  PAYMENT\_AMOUNT,PAYED\_AMOUNT, PAYMENT\_METHOD,  DUE\_DATE, PAYMENT\_STATUS,PRODUCT\_CODE,  UPDATE\_TIMESTAMP, INSERT\_TIMESTAMP  FROM CRDT.PAYMENT\_ORDER  WHERE DUE\_DATE=CURRENT\_DATE; |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve all Direct Debit Credit transactions from table CRDT.PAYMENT\_ORDER for the current date and insert them into History table CRDT.PAYMENT\_ORDER\_HIS. After the insertion delete them from CRDT.PAYMENT\_ORDER |
| **PROGRAM MAIN LOOP** | The main steps performed are:   * Open Cursor against table CRDT.PAYMENT\_ORDER for transactions of the Current Date * Fetch first row and in a loop till the end of Cursor   + Insert row into History table   + Delete row from table   + COMMIT work after every 100 rows   + Fetch next row * When out of Cursor with SQLCODE = 100   + Close Cursor   + End of Program |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.PAYMENT\_ORDER  CRDT.PAYMENT\_ORDER\_HIS |

## 

## [Massive Loading ETE]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Massive Loading ETE |
| **Batch Flow Title** | KRPPD032 - KRPPD033 - KRPPD034 |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | Way4 |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   Massive Loading ETE is being used either for bank executives or for promotional reasons. It uses deposits services for deblocking and debiting the loading amount as well as client services for validations purposes. |

### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KRPPD032] Massive Loading ETE request Processing

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD032** | KV42P | KATP.PD.KEMPC.FTIP.KF6PPINP.SEQ | CRDP.PD.MASLO.FTOP.CRD03201.SEQ | This step reads the file from the NBG staff application and sets up a file to be sent to WAY4. There it will be supplemented with SYDIPEL, card number, and event code. |
|  | KV42P | CRDP.PD.MASLO.FTIP.CRD03202.SEQ | CRDP.PD.MASLO.FTOP.CRD03202.SEQ | This step reads the file like NBG staff application and sets up a file to be sent to WAY4. There it will be supplemented with SYDIPEL, card number, and event code. |
|  | CLRSEQ | CRDP.PD.MASLO.FTIP.CRD03202.SEQ | N/A | Clear file |

#### 

#### [KRPPD033] Massive Loading ETE request Processing

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD033** | KV43P | CRDP.PD.MASLO.FTIP.CRD03301.SEQ | CRDP.PD.MASLO.FTOP.CRD03301.SEQ,  CRDP.PD.MASLO.SEMT.CRD03301.SEQ | Hold loading amount to input file’s account and send file to WAY4 to charge cards (NBG staff). |
|  | KV43P | CRDP.PD.MASLO.FTIP.CRD03302.SEQ | CRDP.PD.MASLO.FTOP.CRD03302.SEQ,  CRDP.PD.MASLO.SEMT.CRD03302.SEQ | Hold loading amount to input file’s account and send file to WAY4 to charge cards (promotional charges). |
|  | KV46P | CRDP.PD.MASLO.SEMT.CRD03301.SEQ,  CRDP.PD.MASLO.SEMT.CRD03302.SEQ | CRDP.PD.MASLO.CMOD.CRD03301.SEQ | Create CMOD data forms regarding SYDIPEL discrepancy. |
|  | IRXJCL | N/A | N/A | Data form of SYDIPEL discrepancy during mass loading. Run:  RUNCMOD CRDP.PD.MASLO.CMOD.CRD03301.SEQ CRD065DPL DIOIKHSHPL |
|  | IEFBR14 | CRDP.PD.MASLO.SEMT.CRD03301.SEQ,  CRDP.PD.MASLO.SEMT.CRD03302.SEQ | N/A | Deallocation of files |

#### [KRPPD034] Massive Loading ETE response Processing

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD034** | KV44P | CRDP.PD.MASLO.FTIP.CRD03401.SEQ | CRDP.PD.MASLO.CMOD.CRD03401.SEQ | Un-hold and credit accounts of successful loadings on WAY4. Create CMOD data forms with unsuccessful loadings (run NBG staff) |
|  | KV44P | CRDP.PD.MASLO.FTIP.CRD03402.SEQ | CRDP.PD.MASLO.CMOD.CRD03402.SEQ | Un-hold and credit accounts of successful loadings on WAY4. Create CMOD data forms with unsuccessful loadings (run promotional loadings) |
|  | SORT | CRDP.PD.MASLO.FTIP.CRD03401.SEQ | KATP.PD.KEMPC.FTOP.KF6PPOUT.SEQ | Create response file for loading NBG staff |
|  | IRXJCL | N/A | N/A | Data form of unsuccessful massive loading VISA Prepaid NBG/Virtual-NBG staff. Run:  RUNCMOD CRDP.PD.MASLO.CMOD.CRD03401.SEQ CRD063DPL DIOIKHSHPL |
|  | IRXJCL | N/A | N/A | Data form of unsuccessful massive loading VISA Prepaid NBG/Virtual-Promotions. Run:  RUNCMOD CRDP.PD.MASLO.CMOD.CRD03402.SEQ CRD064DPL DIOIKHSHPL |

#### [KV42P] Massive Loading of VISA Prepaid NBG - NBG staff application

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | KV42PIN1, File including cards and accounts for massive loading  DCL 1 RKV42PIN1,  2 ACC\_TYPE CHAR(04) INIT(' '),  2 FIL1 CHAR(01) INIT(' '),  2 IFLD\_ID CHAR(20) INIT(' '),  2 FIL2 CHAR(01) INIT(' '),  2 TRANS\_AMOUNT PIC'(13)9V.99' INIT(0),  2 FIL3 CHAR(01) INIT(' '),  2 SAP\_ACCOUNT CHAR(27) INIT(' '),  2 FIL4 CHAR(01) INIT(' '),  2 RECONCILIATION\_ID CHAR(20) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | KV42POU1, file to WAY4 for card and IP addition  DCL 1 RKV42POU1,  2 ROWCODE CHAR(002) INIT(' '),  2 REST\_DETAILS CHAR(130) INIT(' ');    DCL 1 FH\_KV42POU1 BASED(ADDR(RKV42POU1)),  2 FH\_ROWCODE CHAR (002) INIT('FH'),  2 FH\_TITLE CHAR (020) INIT('MASS\_LOADING\_PRD'),  2 FH\_PROC CHAR (010) INIT(' '),  2 FH\_DATE CHAR (010) INIT('01.01.0001'),  2 FH\_SEQ\_NUM CHAR (003) INIT(' '),  2 FH\_FILLER CHAR (087) INIT(' ');    DCL 1 FD\_KV42POU1 BASED(ADDR(RKV42POU1)),  2 FD\_ROWCODE CHAR (002) INIT('FD'),  2 FD\_ACCOUNT\_TYPE CHAR(004) INIT(' '),  2 FD\_FILLER CHAR(001) INIT(' '),  2 FD\_CARD\_NUM CHAR (016) INIT(' '),  2 FD\_IP PIC'(11)9' INIT(0),  2 FD\_CARD\_ID CHAR (020) INIT(' '),  2 FD\_SAP\_ACCOUNT CHAR (027) INIT(' '),  2 FD\_LOADING\_AMOUNT PIC'(13)9V.99' INIT(0),  2 FD\_CLSF\_FIRST\_LOAD CHAR (001) INIT(' '),  2 FD\_EVENT\_CODE CHAR (002) INIT('00'),  2 FD\_RECONC\_KEY CHAR (032) INIT(' ');    DCL 1 FT\_KV42POU1 BASED(ADDR(RKV42POU1)),  2 FT\_ROWCODE CHAR (002) INIT('FT'),  2 FT\_REC\_COUNT PIC'(07)9' INIT(0),  2 FT\_FILLER CHAR (123) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | KV42PIN1\_EOF: Raised when end of file is reached |
| **SERVICES** | N/A |
| **Main Process Logic** | A simple program, where headers are written in the output file, and then with the data from every like of input file we add a row on the output file (some values are static). Finally, file trailer is added in output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * Account type, IFLD ID, SAP account, transaction amount and reconciliation ID values are taken from the input file, with additional static values to be written in output file |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

#### [KV43P] Massive Loading VISA/Mastercard Prepaid NBG - Request to WAY4

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | KV43PIN1: File including cards for massive loading of VISA/Mastercard Prepaid NBG  DCL 1 RKV43PIN1,  2 ROWCODE CHAR(002),  2 REST\_DETAILS CHAR(130);    DCL 1 FH\_KV43PIN1 BASED(ADDR(RKV43PIN1)),  2 FH\_ROWCODE CHAR (002),  2 FH\_TITLE CHAR (020),  2 FH\_PROC CHAR (012),  2 FH\_DATE CHAR (010),  2 FH\_SEQ\_NUM CHAR (003),  2 FH\_FILLER CHAR (085);    DCL 1 FD\_KV43PIN1 BASED(ADDR(RKV43PIN1)),  2 FD\_ROWCODE CHAR (002),  2 FD\_ACCOUNT\_TYPE CHAR(004),  2 FD\_FILLER CHAR(001),  2 FD\_CARD\_NUM CHAR (016),  2 FD\_IP PIC'(11)9',  2 FD\_CARD\_ID CHAR (020),  2 FD\_SAP\_ACCOUNT CHAR(027),  2 FD\_LOADING\_AMOUNT PIC'(13)9V.99',  2 FD\_CLSF\_FIRST\_LOAD CHAR (001),  2 FD\_EVENT\_CODE CHAR (002),  2 FD\_RECONC\_KEY CHAR (032) INIT(' ');  DCL 1 FT\_KV43PIN1 BASED(ADDR(RKV43PIN1)),  2 FT\_ROWCODE CHAR (002),  2 FT\_REC\_COUNT PIC'(07)9',  2 FT\_FILLER CHAR (123); |
| **OUTFILE1 FILE DECLARATION** | KV43POU1: File including response to WAY4  DCL 1 RKV43POU1,  2 ROWCODE CHAR(002) INIT(' '),  2 REST\_DETAILS CHAR(240) INIT(' ');    DCL 1 FH\_KV43POU1 BASED(ADDR(RKV43POU1)),  2 FH\_ROWCODE CHAR (002) INIT('FH'),  2 FH\_TITLE CHAR (020) INIT('MASS\_LOADING\_PRD'),  2 FH\_PROC CHAR (010) INIT(' '),  2 FH\_DATE CHAR (010) INIT('01.01.0001'),  2 FH\_SEQ\_NUM CHAR (003) INIT(' '),  2 FH\_FILLER CHAR (197) INIT(' ');    DCL 1 FD\_KV43POU1 BASED(ADDR(RKV43POU1)),  2 FD\_ROWCODE CHAR (002) INIT('FD'),  2 FD\_ACCOUNT\_TYPE CHAR (004) INIT(' '),  2 FD\_FILLER CHAR (001) INIT(' '),  2 FD\_CARD\_NUM CHAR (016) INIT(' '),  2 FD\_IP PIC'(11)9' INIT(0),  2 FD\_CARD\_ID CHAR (020) INIT(' '),  2 FD\_SAP\_ACCOUNT CHAR(027) INIT(' '),  2 FD\_LOADING\_AMOUNT PIC'(13)9V.99' INIT(0),  2 FD\_COMMISSION PIC'(13)9V.99' INIT(0),  2 FD\_CLSF\_FIRST\_LOAD CHAR (001) INIT(' '),  2 FD\_EVENT\_CODE CHAR (002) INIT(' '),  2 FD\_RRN CHAR (012) INIT(' '),  2 FD\_RECONC\_KEY CHAR (032) INIT(' '),  2 FD\_REASON\_OF\_FAILURE CHAR (080) INIT(' '),  2 FD\_RETURN\_CODE CHAR (002) INIT(' ');  DCL 1 FT\_KV43POU1 BASED(ADDR(RKV43POU1)),  2 FT\_ROWCODE CHAR (002) INIT('FT'),  2 FT\_REC\_COUNT PIC'(07)9' INIT(0),  2 FT\_FILLER CHAR (233) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | KV43POU2: File for CMOD for SYDIPEL disagreements  DCL 1 DETAIL\_LINE\_1,  5 BARCODE CHAR(20) INIT(' '),  5 CARD\_NUMBER CHAR(16) INIT(' '),  5 IP\_W4 FIXED DEC(11,0) INIT(0),  5 IP\_CRAT FIXED DEC(11,0) INIT(0); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | KV43PIN1\_EOF: Raised when end of file is reached |
| **SERVICES** | SRVSIBA: Find Account from IBAN  SRVCDES: Hold amount to account  SRVIXAS: Check account-customer relation  SRVIXAL: Check account-customer relation  SRVIXAE: Check for existence of deposit account for customer  SRVSMAS: Check for paid off account  SELORCN: Check if hold action is allowed  SRVAVBA: Check available balance  KCTINS: Insert to KCCT.CRCA\_TPAYMENTS  KCTSEL1: Select from KCCT.CRCA\_TPAYMENTS  KCTUPD1: Update into KCCT.CRCA\_TPAYMENTS  SRVCOMW: Check commission  SRVSACD: Check accounting date |
| **Main Process Logic** | First, the program sets up date and time information for CMOD. Validates input file based on row code:   * FH (file header): checks for header * FD (detail record): update FD\_CLSF\_FIRST\_LOAD from ‘ ‘ to ‘Y’ and event code from ‘ ‘/’00’/’ 0’ to ‘10’) * FT (file trailer): compare number of entries with count in trailer * Any other code is invalid   Write header to output file1, and then loop through every detail record on the input file:  For records with valid card number, create RRN, find SYDIPEL, and if the account’s balance allows it process the transaction and add it to the table.  Finally, write trailer to output file1. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every detail record (RKV43PIN1.ROWCODE = 'FD') of input file:   * **If** no card number is found (FD\_KV43PIN1.FD\_CARD\_NUM = ' '), **then** set the error flag, and response code/response code text. * **If** card number is found (error flag not set) **then**    + Create RRN, and get SYDIPEL (@SRVIXAL.ODATA.CIP.I\_IP) (by calling SRVIXAL).   For cards connected to a different SYDIPEL, add line to CMOD output file (output file2).  **If** response code 4 is returned, **then** for Greek SAP(SUBSTR(FD\_KV43PIN1.FD\_SAP\_ACCOUNT,1,2) = 'GR') Promotional (PARM = 'PROMO') account check if account exists (by calling SRVSIBA).   * **If** error flag is not set, **then** for a GR SAP (SUBSTR(FD\_KV43PIN1.FD\_SAP\_ACCOUNT,1,2) = 'GR') account, **if** loading amount + commission don’t exceed the available balance **then** insert into KCCT.CRCA\_TPAYMENTS (by calling KCTINS), **otherwise** set error flag, and response code/response code text. * Update counters according to error flag value, and write data to output file1 |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | KCCT.CRCA\_TPAYMENTS |

#### [KV46P] Massive Loading VISA Prepaid NBG/Virtual MC - CMOD report

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | INPFILE1: File with cases of dissagreement  DCL 1 RINPFILE1,  2 BARCODE CHAR(20),  2 CARD CHAR(16),  2 I\_IP\_W4 FIXED DEC(11,0),  2 I\_IP\_CRAT FIXED DEC(11,0); |
| **OUTFILE1 FILE DECLARATION** | OUTFILE1: CMOD report  DCL 1 DETAIL\_LINE\_1,  5 \* CHAR(03) INIT(' '),  5 AA CHAR(03) INIT(' '),  5 \* CHAR(02) INIT(' '),  5 BARCODE CHAR(20) INIT(' '),  5 \* CHAR(02) INIT(' '),  5 CARD\_NUMBER CHAR(16) INIT(' '),  5 \* CHAR(02) INIT(' '),  5 IP\_W4 CHAR(11) INIT(' '),  5 \* CHAR(06) INIT(' '),  5 IP\_CRAT CHAR(11) INIT(' '),  5 \* CHAR(57) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | INPFILE1\_EOF: Raised when end of file is reached |
| **SERVICES** | N/A |
| **Main Process Logic** | First, the program sets up date and time information for CMOD. Program reads through input file and write info to output file, including header and detail rows |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * Add a header every (about) 45 lines * Add data row based on input file data: AA (counter), Barcode, Card Number, SYDIPEL (WAY4), SYDIPEL |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

#### [KV44P] Massive Loading of VISA Prepaid NBG - response by WAY4

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | KV44PIN1: File including cards and accounts for massive loading  DCL 1 RKV44PIN1,  2 ROWCODE CHAR(002) INIT(' '),  2 REST\_DETAILS CHAR(241) INIT(' ');    DCL 1 FH\_KV44PIN1 BASED(ADDR(RKV44PIN1)),  2 FH\_ROWCODE CHAR (002) INIT('FH'),  2 FH\_TITLE CHAR (020) INIT(' '),  2 FH\_PROC CHAR (010) INIT(' '),  2 FH\_DATE CHAR (010) INIT('01.01.0001'),  2 FH\_SEQ\_NUM CHAR (003) INIT(' '),  2 FH\_FILLER CHAR (198) INIT(' ');    DCL 1 FD\_KV44PIN1 BASED(ADDR(RKV44PIN1)),  2 FD\_ROWCODE CHAR (002) INIT('FD'),  2 FD\_ACCOUNT\_TYPE CHAR (004) INIT(' '),  2 FD\_FILLER CHAR (001) INIT(' '),  2 FD\_CARD\_NUM CHAR (016) INIT(' '),  2 FD\_IP PIC'(11)9' INIT(0),  2 FD\_CARD\_ID CHAR (020) INIT(' '),  2 FD\_SAP\_ACCOUNT CHAR(027) INIT(' '),  2 FD\_LOADING\_AMOUNT PIC'(13)9V.99' INIT(0),  2 FD\_COMMISSION PIC'(13)9V.99' INIT(0),  2 FD\_CLSF\_FIRST\_LOAD CHAR (001) INIT(' '),  2 FD\_EVENT\_CODE CHAR (002) INIT(' '),  2 FD\_RRN CHAR (012) INIT(' '),  2 FD\_ARN CHAR (032) INIT(' '),  2 FD\_REASON\_OF\_FAILURE CHAR (080) INIT(' '),  2 FD\_RETURN\_CODE CHAR (002) INIT(' '),  2 FD\_UPD\_FLG CHAR (001) INIT(' ');    DCL 1 FT\_KV44PIN1 BASED(ADDR(RKV44PIN1)),  2 FT\_ROWCODE CHAR (002) INIT('FT'),  2 FT\_REC\_COUNT PIC'(07)9' INIT(0),  2 FT\_FILLER CHAR (234) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | KV44POU1: CMOD data form with rejected cards of NBG staff  DCL 1 DETAIL\_LINE\_1,  5 \* CHAR(02) INIT(' '),  5 AA CHAR(04) INIT(' '),  5 \* CHAR(01) INIT(' '),  5 BARCODE CHAR(20) INIT(' '),  5 \* CHAR(02) INIT(' '),  5 LOADING\_AMOUNT CHAR(16) INIT(' '),  5 \* CHAR(02) INIT(' '),  5 SAP\_ACCOUNT CHAR(27) INIT(' '),  5 \* CHAR(02) INIT(' '),  5 RECONCILIATION\_ID CHAR(20) INIT(' '),  5 \* CHAR(01) INIT(' '),  5 REASON\_TEXT CHAR(30) INIT(' '),  5 \* CHAR(02) INIT(' '),  5 RETURN\_CODE CHAR(02) INIT(' '),  5 \* CHAR(02) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | KV44PIN1\_EOF: Raised when end of file is reached |
| **SERVICES** | SRVSIBA: Find account from IBAN  SRVLAC3: Credit amount to account  SCCTSEL: Get commission from CCS  SRVDDES: Unhold amount on account  KCTSEL1: Select from KCCT.CRCA\_TPAYMENTS  KCTUPD1: Update payment in KCCT.CRCA\_TPAYMENTS  KILSACD: Check accounting date  SRVCOMW: Check commission  SRVCKSF: Check for NBG employee  SRVIXAS: Check account-customer relation  SRVIXAL: Check account-customer relation |
| **Main Process Logic** | First, program sets up date and time for CMOD. Validates input file based on row code:   * FH (file header): checks for header * FD (detail record): set total amount * FT (file trailer): compare number of entries with count in trailer * Any other code is invalid   Loop through every detail record on the input file:  Calculate commission and update CSS accordingly if card is loaded, Write data to output file, and when appropriate update transactions on table. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every detail record (RKV44PIN1.ROWCODE = 'FD') of input file:   * **If** no card number found, **then** rewrite input file accordingly, and set update flag to ‘9’. * For update flag set to ‘1’, simply update counter * For update flag set to ‘9’:   + For NBG staff (PARM = 'STAFF'), write data to output file (plus add header every (about) 45 lines: AA, Barcode, Loading amount, SAP account, reconciliation ID, return code/text.   + For Promotional (PARM = 'PROMO'), write data to output file (plus add header every (about) 45 lines: AA, Barcode, Loading amount, SAP account, reconciliation ID, * For update flag set to ‘ ‘:   + Get SYDIPEL (by calling SRVIXAL)   + **If** no error has appeared so far (FD\_KV44PIN1.FD\_RETURN\_CODE = '00' & L\_ERR\_FLG = '0'), **then** for GR SAP accounts (SUBSTR(FD\_KV44PIN1.FD\_SAP\_ACCOUNT,1,2) = 'GR'):     - credit loading amount (and commission if applicable) to account and accounting (by calling SRVLAV3)     - update INT\_KEY in KCCT.CRCA\_TPAYMENTS (by calling KCTUPD1)   + For GR SAP accounts (SUBSTR(FD\_KV44PIN1.FD\_SAP\_ACCOUNT,1,2) = 'GR'):     - **If** error flag isn’t set **then** check for commission (by calling SCCTSEL), and **if** it gets added (@SCCTSEL.RC = 0), **then**        * **If** commission has not been collected (@SCCTSEL.ODATA.COLLECTION\_COUNT = 0 & @SCCTSEL.ODATA.COLLECTION\_SUM = 0.0) **then** check commission (collection) (by calling SRVCOMW)       * **Otherwise** set error flag to 1, and set response code/text     - **If** error flag isn’t set, **then** set update flag to 1 and update input file     - **Otherwise** (if it is set) set update flag to 9, and if no error is shown by return code then update input file   + For GR SAP accounts (SUBSTR(FD\_KV44PIN1.FD\_SAP\_ACCOUNT,1,2) = 'GR'), **if** return code is set to 99 (FD\_KV44PIN1.FD\_RETURN\_CODE = '99'), **then** check commission (collection cancellation of registration without collection) (by calling SRVCOMW)   + Set error flag to 1   + **If** error flag is set and return code is 00, **then** update input file   + For NBG staff (PARM = 'STAFF'), write data to output file (plus add header every (about) 45 lines: AA, Barcode, Loading amount, SAP account, reconciliation ID, return code/text.   + For Promotional (PARM = 'PROMO') write data to output file (plus add header every (about) 45 lines: AA, Barcode, Loading amount, SAP account, reconciliation ID,   + Commit DB2 action * Otherwise no action. |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | KCCT.CRCA\_TPAYMENTS |

## 

## [Massive (Un)Loading Voucher]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Massive Loading Voucher |
| **Batch Flow Title** | KRPPD013-015 & KRPPD017-019 |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | Way4 |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   The Massive (Un)Loading Voucher process uses a flat file for every organization’s voucher cards. It uses a parametric card table as well as deposit services for debit(credit) and client services for validations. |

### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KRPPD013] Massive Loading Voucher request Processing

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD013** | SORT | CRDP.PD.PRVOU.INTT.K95CPOUT.SEQ | CRDP.PD.PRVOU.SEMT.CRD01302.SEQ | Sort input file to only get the header. |
|  | SORT | CRDP.PD.PRVOU.INTT.K95CPOUT.SEQ | CRDP.PD.PRVOU.SEMT.CRD01303.SEQ | Sort input file to get only IBANS. |
|  | SORT | CRDP.PD.PRVOU.SEMT.CRD01303.SEQ | CRDP.PD.PRVOU.SEMT.CRD01304.SEQ | Remove duplicate IBANs from the previous step. |
|  | K75CP | CRDP.PD.PRVOU.SEMT.CRD01302.SEQ,  CRDP.PD.PRVOU.SEMT.CRD01304.SEQ | CRDP.PD.PRVOU.SEMT.CRD01305.SEQ | Merge “header” and “iban” files from previous steps. |
|  | K83CP | CRDP.PD.PRVOU.SEMT.CRD01305.SEQ | CRDP.PD.PRVOU.INTT.CRD01306.SEQ | Enrich input file with validations rc and SYDIPEL. |
|  | IEFBR14 | CRDP.PD.PRVOU.SEMT.CRD01302.SEQ,  CRDP.PD.PRVOU.SEMT.CRD01303.SEQ,  CRDP.PD.PRVOU.SEMT.CRD01304.SEQ,  CRDP.PD.PRVOU.SEMT.CRD01305.SEQ | N/A | Deallocation of files |

#### 

#### [KRPPD014] Massive Loading Voucher request Processing

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD014** | SORT | CRDP.PD.PRVOU.INTT.K95CPOUT.SEQ | CRDP.PD.PRVOU.SEMT.CRD01401.SEQ | Sort input file, to only get details. |
|  | SORT | CRDP.PD.PRVOU.INTT.CRD01306.SEQ,  CRDP.PD.PRVOU.SEMT.CRD01401.SEQ | CRDP.PD.PRVOU.SEMT.CRD01402.SEQ | Join 2 files. |
|  | K84CP | CRDP.PD.PRVOU.SEMT.CRD01402.SEQ | CRDP.PD.PRVOU.SEMT.CRD01403.SEQ | Validate input file, and enrich with fee. |
|  | IEFBR14 | CRDP.PD.PRVOU.INTT.CRD01306.SEQ,  CRDP.PD.PRVOU.SEMT.CRD01401.SEQ,  CRDP.PD.PRVOU.SEMT.CRD01402.SEQ,  CRDP.PD.PRVOU.SEMT.CRD01403.SEQ | N/A | Deallocation of files. |

#### [KRPPD015] Massive Loading Voucher request Processing

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD015** | DSNTIAUL | N/A | CRDP.PD.PRVOU.SEMT.CRD01501.SEQ | Unload CRDT.PREPAID\_REQ\_LOA for sum amounts. |
|  | K74CP | CRDP.PD.PRVOU.SEMT.CRD01501.SEQ | N/A | Place holds on accounts based on total amount and fee from input file. |
|  | DSNTIAUL | N/A | CRDP.PD.PRVOU.SEMT.CRD01502.SEQ | Unload CRDT.PREPAID\_REQ\_LOA for status 1. |
|  | SORT | CRDP.PD.PRVOU.SEMT.CRD01502.SEQ | CRDP.PD.PRVOU.FTOP.CRD01503.SEQ | Copy to permanent FTP file and enrich file with header and trailer. |
|  | IEFBR14 | CRDP.PD.PRVOU.SEMT.CRD01501.SEQ,  CRDP.PD.PRVOU.SEMT.CRD01502.SEQ | N/A | Deallocation of files |

#### [K75CP] Merge Heder-IBANs for (Un)Loading request

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | INPFILE1: Header file  DCL 1 I\_RECORD\_1,  5 HEADER CHAR(25) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | INPFILE1: Distincts IBANs file  DCL 1 I\_RECORD\_2,  5 IBAN CHAR(27) INIT(' '),  5 RC CHAR(02) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | OUTFILE1: Concatenated flat file  DCL 1 O\_RECORD,  5 CONCAT\_FIELD CHAR(54) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_INPUT1: Raised when end of file1 is reached  MORE\_INPUT2: Raised when end of file2 is reached |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve current date.  For every line of the 2nd input file: Write row to output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * Create output file line by concatenating header, IBAN and RC and write to file |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

#### [K83CP] Enrich input file with fee

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | INPFILE1:  DCL 1 I\_RECORD,  5 BUSINESS\_DATE CHAR(10) INIT(' '),  5 TOTAL\_REQS\_CNT PIC'(7)9' INIT(0),  5 SEQ\_NO CHAR(3) INIT(' '),  5 TYPE CHAR(2) INIT(' '),  5 ORG\_CDE CHAR(3) INIT(' '),  5 IBAN CHAR(27) INIT(' '),  5 RC CHAR(02) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | OUTFILE1: Input file enriched with validations rc and SYDIPEL  DCL 1 O\_RECORD,  5 BUSINESS\_DATE CHAR(10) INIT(' '),  5 TOTAL\_REQS\_CNT PIC'(7)9' INIT(0),  5 SEQ\_NO CHAR(3) INIT(' '),  5 TYPE CHAR(2) INIT(' '),  5 ORG\_CDE CHAR(3) INIT(' '),  5 LE\_SYDIPEL PIC'(11)9' INIT(0),  5 IBAN CHAR(27) INIT(' '),  5 LE\_DEP\_ACC CHAR(16) INIT(' '),  5 REQUEST\_STATUS CHAR (001) INIT('0'),  5 RC CHAR (002) INIT(' '),  5 REJECTION\_TXT CHAR(80) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_INPUT1: Raised when end of file is reached |
| **SERVICES** | SRVIXAS: Check account-customer relation  SRVSIBA: Find Account from IBAN  SRVCKCS: Find customers connect to card, by card number  ICDPCM1: Insert record to CRDT.PREPAID\_REQ |
| **Main Process Logic** | Retrieve current date.  Retrieve institution deposit account via IBAN (by calling SRVSIBA). If possible (L\_INST\_DEP\_ACC ^= 0), retrieve company SYDIPEL via account.  For every line of the input file: Validate client data/ Insert data to table, and write data to output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * **If** record has not been rejected (I\_RECORD.RC = '00'), **then**    + **If** organization code (I\_RECORD.ORG\_CDE) is not ‘ETE’ (NBG) **then** validate client data (by calling SRVCKCS)   + **Else** insert data into CRDT.PREPAID\_REQ (by calling ICDPCM1) * Write input to output file |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.PREPAID\_REQ |

#### [K84CP] Validate and enrich input file with fee

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | INPFILE1:  DCL 1 I\_RECORD,  5 BUSINESS\_DATE CHAR(10) INIT(' '),  5 SEQ\_NO CHAR(3) INIT(' '),  5 TYPE CHAR(2) INIT(' '),  5 ORG\_CDE CHAR(3) INIT(' '),  5 CARD\_ID CHAR(20) INIT(' '),  5 LOADING\_AMOUNT PIC'(13)9V.99' INIT(0),  5 LE\_DEP\_ACC CHAR(16) INIT(' '),  5 LE\_SYDIPEL PIC'(11)9' INIT(0),  5 IBAN CHAR(27) INIT(' '),  5 REQUEST\_STATUS CHAR (1) INIT('0'),  5 RC CHAR (2) INIT(' '),  5 REJECTION\_TXT CHAR(80) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | OUTFILE1:  DCL 1 O\_RECORD,  05 BUSINESS\_DATE CHAR(10) INIT(' '),  05 SEQ\_NO CHAR(3) INIT(' '),  05 TYPE CHAR(2) INIT(' '),  05 ORG\_CDE CHAR(3) INIT(' '),  05 CARD\_ID CHAR(20) INIT(' '),  05 LE\_SYDIPEL PIC'(11)9' INIT(0),  05 LE\_DEP\_ACC CHAR(16) INIT(' '),  05 REQUEST\_STATUS CHAR(2) INIT(' '),  05 AMOUNT PIC'(13)9V.99' INIT(0),  05 FEE PIC'(13)9V.99' INIT(0),  05 COMM\_LOG\_ID CHAR(26) INIT(' '),  05 IBAN CHAR(27) INIT(' '),  05 RC CHAR(2) INIT(' '),  05 REJECTION\_TXT CHAR(80) INIT(' '),  05 INSERT\_TMSTMP CHAR(26) INIT('0001-01-01-00.00.00.000000'); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_INPUT1: Raised when end of file is reached |
| **SERVICES** | SCCSCAL: Calculate commission  SRVIIPS: Get customer information  ICDLOA1: Insert into CRDT.PREPAID\_REQ\_LOA  SCCCPN2: Get data from KCCT.CARDS\_PIN  SRVIXAS: Check account-customer relation  SRVONBL: Check youth onboarding |
| **Main Process Logic** | For every line of the input file: According to the case, retrieve necessary customer/card information, calculate commission, and perform validations. Write data to output file. Insert data to CRDT.PREPAID\_REQ\_LOA. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * Store unique identification data * For record response code (RC) ‘00’/’0’/’ ‘/’ ‘:   + Retrieve current date   + For request status ‘0’:     - Convert 2-digit type to full type and generate prod event     - Retrieve customer’s branch (by calling SRVIIPS)     - Calculate commission per card (by calling SCCSCAL)     - For record type unload (I\_RECORD.TYPE = 'UL'):       * **If** commission fee (@SCCSCAL.ODATA.COL\_AMNT) is greater than the loading amount (I\_RECORD.LOADING\_AMOUNT), **then** set it to loading amount     - For record type load (I\_RECORD.TYPE = 'LD'):       * Retrieve card based in CARD\_ID (bu calling SCCCPN2)       * Retrieve SYDIPEL based on card (by calling SRVIXAS)       * Check youth onboarding (by calling SRVONBL)   + Otherwise set RC/TXT * Otherwise set RC/TXT * Write data to output file * Insert details into CRDT.PREPAID\_REQ\_LOA (by calling ICDLOA1) |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.PREPAID\_REQ\_LOA  KCCT.CARDS\_PIN |

#### [K74CP] Place hold on accounts

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | INPFILE1:  DCL 1 I\_RECORD,  5 BUSINESS\_DATE CHAR(10) INIT(' '),  5 SEQ\_NO CHAR(3) INIT(' '),  5 TYPE CHAR(2) INIT(' '),  5 ORG\_CDE CHAR(3) INIT(' '),  5 LE\_DEP\_ACC CHAR(16) INIT(' '),  5 SUM\_AMOUNT PIC'(13)9V.99' INIT(0),  5 FILLER1 CHAR(1) INIT(''),  5 DEB\_AMOUNT CHAR(1) INIT(' '),  5 SUM\_FEE PIC'(13)9V.99' INIT(0),  5 FILLER2 CHAR(1) INIT(''),  5 DEB\_FEE CHAR(1) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_INPUT1: Raised when end of file is reached |
| **SERVICES** | UCDPCM1: Update record in CRDT.PREPAID\_REQ  UCDLOA: Update record in CRDT.PREPAID\_REQ\_LOA  UCDISS1: Update record in CRDT.PREPAID\_REQ\_ISS  SRVCDES: Hold amount on account  SRVAVBA: Calculate available balance |
| **Main Process Logic** | Get current date. Store unique identification data.  For every line of the input file: Calculate amount and fee for every record, and update data in tables. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * Calculate available balance per account (by calling SRVAVBA) * Calculate hold parameters:   + **If** DEB\_AMOUNT ‘D’ and non-zero sum amount (I\_RECORD.SUM\_AMOUNT!=0) **then**     - Calculate amount, and **if** it doesn’t exceed the available balance **then** hold amount (by calling SRVCDES)     - Else (if it does exceed the available balance) update record in CRDT.PREPAID\_REQ (by calling UCDPCM1)     - Update record in CRDT.PREPAID\_REQ\_LOA/ISS (by calling UCDLOA/UCDISS1)   + **If** DEB\_FEE ‘D’ and non-zero sum fee (I\_RECORD.SUM\_FEE!=0) **then**     - Calculate fee, and **if** it doesn’t exceed the available balance or it is Unload **then** hold fee(by calling SRVCDES)     - Else (if it does exceed the available balance) update record in CRDT.PREPAID\_REQ (by calling UCDPCM1)     - Update record in CRDT.PREPAID\_REQ\_LOA/ISS (by calling UCDLOA/UCDISS1)   + Else If DEB\_FEE ‘D’ and no sum fee (I\_RECORD.SUM\_FEE=0)     - For Request Status 0:       * Update record in CRDT.PREPAID\_REQ (by calling UCDPCM1) as success       * Update record in CRDT.PREPAID\_REQ\_LOA/ISS (by calling UCDLOA/UCDISS1)     - Otherwise:       * Update record in CRDT.PREPAID\_REQ (by calling UCDPCM1) as fail       * Update record in CRDT.PREPAID\_REQ\_LOA/ISS (by calling UCDLOA/UCDISS1 |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.PREPAID\_REQ  CRDT.PREPAID\_REQ\_LOA  CRDT.PREPAID\_REQ\_ISS |

#### [KRPPD017] Massive Loading Voucher Response Processing

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD017** | SORT | CRDP.PD.PRVOU.FTIP.CRD01701.SEQ | CRDP.PD.PRVOU.SEMT.CRD01702.SEQ | Remove header, trailer, and REC\_TYPE from input file. |
|  | SORT | CRDP.PD.PRVOU.SEMT.CRD01702.SEQ | CRDP.PD.PRVOU.SEMT.CRD01703.SEQ | Remove header, trailer, and REC\_TYPE from input file. |
|  | SORT | CRDP.PD.PRVOU.SEMT.CRD01703.SEQ | CRDP.PD.PRVOU.SEMT.CRD01700.SEQ | Build Querry, check if keys exist in history table. |
|  | DSNTIAUL | CRDP.PD.PRVOU.SEMT.CRD01700.SEQ | CRDP.PD.PRVOU.SEMT.CRD017A0.SEQ | Search keys in CRDT.PREPAID\_REQ\_LOA\_HIST |
|  | IDCAMS | CRDP.PD.PRVOU.SEMT.CRD017A0.SEQ | N/A | Abend if key exists |
|  | K76CP | CRDP.PD.PRVOU.SEMT.CRD01703.SEQ | N/A | Update CRDT.PREPAID\_REQ\_LOA |
|  | IEFBR14 | CRDP.PD.PRVOU.SEMT.CRD01702.SEQ,  CRDP.PD.PRVOU.SEMT.CRD01703.SEQ,  CRDP.PD.PRVOU.SEMT.CRD01700.SEQ,  CRDP.PD.PRVOU.SEMT.CRD017A0.SEQ | N/A | Deallocation of files |

#### 

#### [KRPPD018] Massive Loading Voucher Response Processing

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD018** | DSNTIAUL | N/A | CRDP.PD.PRVOU.SEMT.CRD01801.SEQ | Unload CRDT.PREPAID\_REQ\_LOA for sum amounts |
|  | K77CP | CRDP.PD.PRVOU.SEMT.CRD01801.SEQ | N/A | Deblock and Debit/Credit account |
|  | IEFBR14 | CRDP.PD.PRVOU.SEMT.CRD01801.SEQ | N/A | Deallocation of files |

#### [KRPPD019] Massive Loading Voucher Response Processing

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD019** | DSNTIAUL | N/A | CRDP.PD.PRVOU.SEMT.CRD01901.SEQ | Unload CRDT.PREPAID\_REQ\_LOA for logging |
|  | K78CP | CRDP.PD.PRVOU.SEMT.CRD01901.SEQ | N/A | Perform validations and deliver an output file to IL |
|  | SORT | CRDP.PD.PRVOU.SEMT.CRD01901.SEQ | CRDP.PD.PRVOU.FTOP.CRD01902.SEQ | Export file to card division |
|  | K87CP | N/A | N/A | Archive all data from CRDT.PREPAID\_REQ\_ISS and CRDT.PREPAID\_REQ\_ISS\_HIST |
|  | IEFBR14 | CRDP.PD.PRVOU.SEMT.CRD01901.SEQ | N/A | Deallocation of files |

#### [K76CP] Update CRDT.PREPAID\_REQ\_LOA

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | INPFILE1:  DCL 1 I\_RECORD,  5 BUSINESS\_DATE CHAR(10) INIT(' '),  5 SEQ\_NO CHAR(3) INIT(' '),  5 TYPE CHAR(2) INIT(' '),  5 ORG\_CDE CHAR(3) INIT(' '),  5 CARD\_ID CHAR(20) INIT(' '),  5 LOADING\_AMOUNT PIC'(13)9V.99' INIT(0),  5 LE\_DEP\_ACC CHAR(16) INIT(' '),  5 LE\_SYDIPEL PIC'(11)9' INIT(0),  5 POSTING\_STATUS CHAR (32) INIT(' '),  5 RC CHAR (2) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_INPUT1: Raised when end of file is reached |
| **SERVICES** | UCDLOA1: Update record in CRDT.PREPAID\_REQ\_LOA |
| **Main Process Logic** | Retrieve current date.  For every line of the input file: Update record in table CRDT.PREPAID\_REQ\_LOA.  Run massive update (in CRDT.PREPAID\_REQ\_LOA) for reject requests. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * For non rejected record (RC=00): set local response code (RC) to 2 * Otherwise set local RC to 7 and set RC TXT according to record type Load/Unload (LD/UL) * Update record in CRDT.PREPAID\_REQ\_LOA (by calling UCDLOA1) |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.PREPAID\_REQ\_LOA |

#### [K77CP] Deblock and Debit/Credit account

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | INPFILE1: Edited MLBAT2 exported file-amount and fee per institution  DCL 1 I\_RECORD,  5 BUSINESS\_DATE CHAR(10) INIT(' '),  5 SEQ\_NO CHAR(3) INIT(' '),  5 TYPE CHAR(2) INIT(' '),  5 ORG\_CDE CHAR(3) INIT(' '),  5 LOADING\_SUM\_AMOUNT PIC'(13)9V.99' INIT(0),  5 FILLER1 CHAR(1) INIT(''),  5 LOADING\_SUM\_FEE PIC'(13)9V.99' INIT(0),  5 FILLER2 CHAR(1) INIT(''),  5 LE\_DEP\_ACC CHAR(16) INIT(' '),  5 LE\_SYDIPEL PIC'(11)9' INIT(0),  5 FILLER3 CHAR(1) INIT(''); |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_INPUT1: Raised when end of file is reached |
| **SERVICES** | SCDPCM1: Select from CRDT.PREPAID\_REQ  UCDPCM2: Update in CRDT.PREPAID\_REQ  UCDLOA2: Update in CRDT.PREPAID\_REQ\_LOA (REQUEST\_STATUS, REJECTION\_TXT, RC)  UCDLOA3: Update in CRDT.PREPAID\_REQ\_LOA (REQUEST\_STATUS)  UCDISS5: Update in CRDT.PREPAID\_REQ\_ISS (REQUEST\_STATUS, RC, REJECTION\_TXT)  UCDISS6: Update in CRDT.PREPAID\_REQ\_ISS (REQUEST\_STATUS)  SRVDDES: Unhold amount on account  SRVLAC5: Credit/Debit deposit account and respective accounting account, without checking available balance  INSCMT: Insert into KCCT.COMMISSIONS\_TRN |
| **Main Process Logic** | Retrieve current date.  For every line of the input file: Fetch keys from master. For hold amount/fee unhold amount, if it exists, and set response code, or otherwise simply set the response code. Update CRDT.PREPAID\_REQ\_LOA and CRDT.PREPAID\_REQ\_ISS, and insert into KCCT.COMMISSIONS\_TRN if needed. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * Fetch keys from master (by calling SCDPCM1). * **If** hold amount is non-zero **then**   + **If** debit amount (L\_DEB\_AMOUNT='D') **then** deblock debit amount (by calling SRVDDES)   + **Else**: set local RC to ‘0 ‘ or first character of local RC+0 * **Else**   + **If** Record type IS/RE and loading sum amount = 0 **then**: set local RC to ‘0 ‘ or first character of local RC+0   + **Else**: set local RC to ‘9 ‘ or first character of local RC+9 * **If** hold fee is non-zero **then**   + For record type IS/RE/UL set record code to ‘0 ‘   + deblock debit fee (by calling SRVDDES) * **Else**   + **If** Record is IS/RE/LD/UL and loading sum fee = 0 **then**: set local RC to ‘0 ‘ or first character of local RC+0   + **Else**: set local RC to ‘9 ‘ or first character of local RC+9 * **If** response code is 00 **then**    + For Record type IS/RE set response code to ‘0 ‘   + Otherwise select from CRDT.PREPAID\_REQ\_LOA   + Insert into KCCT.COMMISSIONS\_TRN (by calling INSCMT)   + Credit/Debit deposit account and respective accounting account (by calling SRVLAC5)   + Update CRDT.PREPAID\_REQ\_LOA for type LD/UL (by calling UCDLOA1/UCDLOA2) in case of FAIL/SUCCESS, and CRDT.PREPAID\_REQ\_ISS for type IS/RE (by calling UCDISS5/UCDISS6) in case of FAIL/SUCCESS. * **Else**   + Update CRDT.PREPAID\_REQ\_LOA for type LD/UL (by calling UCDLOA1), and CRDT.PREPAID\_REQ\_ISS for type IS/RE (by calling UCDISS5). |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.PREPAID\_REQ CRDT.PREPAID\_REQ\_LOA  CRDT.PREPAID\_REQ\_ISS  KCCT.COMMISSIONS\_TRN |

#### [K78CP] Perform validations and deliver an output file to IL

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | INPFILE1: From CRDT.PREPAID\_REQ\_LOA/ISS unloading  DCL 1 I\_RECORD,  5 BUSINESS\_DATE CHAR(10) INIT(' '),  5 SEQ\_NO CHAR(3) INIT(' '),  5 TYPE CHAR(2) INIT(' '),  5 ORG\_CDE CHAR(3) INIT(' '),  5 CARD\_ID CHAR(20) INIT(' '),  5 SUM\_AMNT CHAR(16) INIT(' '),  5 IBAN CHAR(27) INIT(' '),  5 REJECTION\_TXT CHAR(80) INIT(' '),  5 RC CHAR(2) INIT(' '),  5 LE\_SYDIPEL CHAR(11) INIT(' '),  5 COM\_LOG\_ID CHAR(26) INIT(' '),  5 FEE FIXED DEC(15,2) INIT(0),  5 LE\_DEP\_ACC CHAR(16) INIT(' '),  5 STATUS CHAR (1) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_INPUT1: Raised when end of file is reached |
| **SERVICES** | UCDLOA1: Update record in CRDT.PREPAID\_REQ\_LOA  SCCSUPD: Commission transaction update (CMFT.SYLOOKUP, CMFT.COMMISSIONLOG, CMFT.COMMISSIONLOGDTL, CMFT.AUTHLOG, CMFT.COMMISSIONLOG\_HIST, CMFT.COMMISSIONLOGDTL\_HIST)  SCCSCNL: Commission transaction cancellation (CMFT.COMMISSIONLOG, CMFT.COMMISSIONLOGDTL) |
| **Main Process Logic** | Retrieve current date.  For every line of the input file: Log successful and unsuccessful requests accordingly. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * For records with fee (I\_RECORD.FEE^=0):   + For record status 4: Log successful requests to NBG (by calling SCCSUPD)   + For record status 6/7/8/9: Log unsuccessful requests to NBG (by calling SCCSCNL)   + Otherwise: do nothing |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CMFT.SYLOOKUP  CMFT.COMMISSIONLOG  CMFT.COMMISSIONLOGDTL  CMFT.AUTHLOG  CMFT.COMMISSIONLOG\_HIST  CMFT.COMMISSIONLOGDTL\_HIST  CMFT.COMMISSIONLOG  CMFT.COMMISSIONLOGDTL |

#### [K87CP] Archive all data from CRDT.PREPAID\_REQ\_ISS and CRDT.PREPAID\_REQ\_ISS\_HIST

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | Insert everything from CRDT.PREPAID\_REQ\_LOA to CRDT.PREPAID\_REQ\_LOA\_HIST.  Delete everything form CRDT.PREPAID\_REQ\_LOA. |
| **PROGRAM MAIN LOOP** | N/A |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.PREPAID\_REQ\_LOA\_HIST  CRDT.PREPAID\_REQ\_LOA |

## 

## [Massive Loading KA]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Massive Loading KA |
| **Batch Flow Title** | KRPPD022 |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | DIAS |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   The Massive Loading KA process uses a flat file for Prepaid KA cards to be loaded. It uses a parametric card table as well as deposit services for debit(credit) and client services for validations. |

### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KRPPD022] Massive Loading KA Processing

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD022** | KS76P | OAFP.PD.PMNTX.INTT.LOADAKEA.SEQ | CRDP.PD.PRKEA.FTOP.CRD02201.SEQ | Massive Loading of Prepaid KA Cards |
|  | IDCAMS | CRDP.PD.PRKEA.FTOP.CRD02201.SEQ | KATP.PD.PRKEA.GDGT.CRD02201.BACK(+1) | Backup the FTOP files before deallocating |
|  | IEFBR14 | OAFP.PD.PMNTX.INTT.LOADAKEA.SEQ | N/A | Deallocation of files |

#### [KS76P] Massive Issuance KA Program

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | KS67PI:  DCL RKS76PI CHAR(37) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | KS67PO:  DCL RKS76PI CHAR(37) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | EOF1: Raised when end of file is reached |
| **SERVICES** | N/A |
| **Main Process Logic** | First, program finds current date and time, and checks for a header in the input file. If no header is found, exit program.  For every other line of the input file: Check if it is detail record of file trailer, and write data to output file.  Check if input file has detail record and file trailer. **If** there is no trailer, **then** exit program.  Check total amount and total record amounts. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * FD (detail record): update lines counter, and total amount * FT (file trailer): compare number of entries with count in trailer * Finally, write data to output file |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

## [Credit Loyalty & Go4More]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Credit Loyalty & Go4More |
| **Batch Flow Title** | KRCPD012 |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | Way4 |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   The clearing process of merchants as toyota or the redemption of go4more points of NBG clients is achieved through the batch process of credit loyalty & Go4more. |

### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KRCPD012] Credit Loyalty & Go4More

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRCPD012** | **K62DP** | CRDP.PD.TOYOT.FTIP.CRD01201.SEQ | CRDP.PD.TOYOT.FTOP.CRD01201.SEQ | CREDIT LOYALTY UPDATES INPUT FOR ROWS PROCESSED ,OUPUT WITH RESULTS |
|  | BACK01 | CRDP.PD.TOYOT.FTOP.CRD01201.SEQ | CRDP.PD.TOYOT.GDGT.CLRRESPF.BACK | CREATE FTOP FILE BACKUP |

#### [K62DP] Credit Loyalty & Go4More

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | CL 1 HRK62DPI\_HEADER BASED(ADDR(RK62DPI)),  2 REC\_TYPE CHAR(02) INIT(' '),  2 CREATED\_DATE CHAR(10) INIT(' '),  2 REC\_NUM PIC '(7)9' INIT('0'),  2 REC\_SUM PIC '(8)9V.99' INIT('0'),  2 FILE\_NAME CHAR(10) INIT(' '),  2 FILLER CHAR(159) INIT(' ');  DCL 1 HRK62DPI BASED(ADDR(RK62DPI)), /\* LENGTH 209 \*/  2 REC\_TYPE CHAR(02) INIT(' '),  2 PROCESS\_DATE CHAR(10) INIT(' '),  2 EXEC\_DATE CHAR(10) INIT(' '),  2 VALUE\_DATE CHAR(10) INIT(' '),  2 OPERATION\_ID CHAR(10) INIT(' '),  2 TRANS\_AMOUNT PIC '(12)9V.99' INIT('0'),  2 TARGET\_ACC CHAR(11) INIT(' '),  2 SOURCE\_ACC CHAR(11) INIT(' '),  2 DEB\_CRE CHAR(01) INIT(' '),  2 REASON\_TXT CHAR(80) INIT(' '),  2 APPL\_REFERENCE CHAR(40) INIT(' '),  2 REASON\_CODE CHAR(03) INIT(' '),  2 CHANNEL CHAR(02) INIT(' '),  2 BRANCH CHAR(03) INIT(' '),  2 CHECK CHAR(01) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | DCL 1 HRK62DPO BASED(ADDR(RK62DPO)), /\* RL=290 \*/  2 REC\_TYPE CHAR(02) INIT(' '),  2 PROCESS\_DATE CHAR(10) INIT(' '),  2 EXEC\_DATE CHAR(10) INIT(' '),  2 VALUE\_DATE CHAR(10) INIT(' '),  2 OPERATION\_ID CHAR(10) INIT(' '),  2 TRANS\_AMOUNT PIC '(12)9V.99' INIT('0'),  2 TARGET\_ACC CHAR(11) INIT(' '),  2 SOURCE\_ACC CHAR(11) INIT(' '),  2 DEB\_CRE CHAR(01) INIT(' '),  2 REASON\_TXT CHAR(80) INIT(' '),  2 APPL\_REFERENCE CHAR(40) INIT(' '),  2 REASON\_CODE CHAR(03) INIT(' '),  2 CHANNEL CHAR(02) INIT(' '),  2 BRANCH CHAR(03) INIT(' '),  2 CHECK CHAR(01) INIT(' '),  2 STATUS CHAR(01) INIT(' '),  2 STATUS\_DESCR CHAR(80) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | @SRVLAC3 Called to DEBIT THE ACCOUNT and Credit the accounting entry  @SRVCDAC Called to Debit / Credit to the linked Debit account |
| **Main Process Logic** | Program execution starts with the following :  ● Get the current Date from Db2 Database  ● Opens Input , Output Files K62DPI , K62DPO  ● Execute the loop until end of file as described below  ● Write the trailer to the Output File  ● Closes Input , Output Files K50DP1 , K50DPO  ● Writes statistics of counters |
| **PROGRAM MAIN LOOP** | ● Write the header for input type HRK62DPI.REC\_TYPE = 'FH' to the Output File same as input  ● Process all the input Lines record by record for HRK62DPI.REC\_TYPE = 'FD'  ● For each input line that has the flag HRK62DPI.CHECK<>'1' different than 1 execute routine SRV\_LINK with the following actions:  ○ Invoke service @SRVCDAC, which is used for credit/debit the client account, using input file parameters  ○ For service call successful RC= 0, SQL Committing work process , update the Input file with flag HRK62DPI.CHECK set to 1 processed , write Output file same as input with extra on HRK62DPO.STATUS= 1 and description HRK62DPO.STATUS\_DESCR = ‘‘  ○ For unsuccessful , update the Input file with flag HRK62DPI.CHECK set to 1 processed  ■ for rc = 21 write Output file same as input with extra on HRK62DPO.STATUS= 2 and description HRK62DPO.STATUS\_DESCR from service return SRVCDAC.RC\_TXT  ■ same with 22,26 and status = 3 , and description SRVCDAC.RC\_TXT, and for any other case status = 4 , and description SRVCDAC.RC\_TXT  ○ Increase counter WS\_READ |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

#### 

## [Massive Issuance KA]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Massive Issuance KA |
| **Batch Flow Title** | KRPPM020 - KRPPD021 |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | DIAS |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   The Massive Issuance KA process uses a flat file from the government for Prepaid KA cards to be created. It uses a parametric card table as well as client services for validations. |

### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KRPPM020] Massive Issuance KA Request Processing

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPM020** | KS94P | OAFP.PD.PMNTX.INTT.CARDSKEA.SEQ  OAFP.PD.PMNTX.INTT.CARDSKEA.SEQ | CRDP.PD.PRKEA.FTOP.CRD02001.SEQ | REQUEST(ISSUING VISA PREPAID KA) |
|  | BACK01 | CRDP.PD.PRKEA.FTOP.CRD02001.SEQ | KATP.PD.PRKEA.GDGT.CRD02001.BACK | BACKUP THE KS94P output FILES BEFORE FTP TO WAY4 |
|  | DEALLOC |  | OAFP.PD.PMNTX.INTT.CARDSKEA.SEQ | DEALLOCATION OF FILES using IBM utility IEFBR14 |

#### [KS94P] Massive Issuance KA Program

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | DCL 1 RHKS94PI BASED(ADDR(RKS94PI)), /\*\_HEADER RECORD\_41 BYTES\*/  2 REC\_TYPE CHAR(02) INIT(' '),  2 ABOUT\_TEXT CHAR(20) INIT(' '),  2 TIMESTAMP CHAR(19) INIT(' ');  DCL 1 RTKS94PI BASED(ADDR(RKS94PI)), /\*\_TRAILER RECORD\_41 BYTES\*/  2 REC\_TYPE CHAR(02) INIT(' '),  2 REC\_COUNT CHAR(07) INIT(' '),  2 FILLER02 CHAR(18) INIT(' ');  DCL 1 RDKS94PI BASED(ADDR(RKS94PI)),  /\*\_\_DETAIL RECORD ΑΝΑΛΥΤΙΚΕΣ ΕΓΓΡΑΦΕΣ \*/  /\*ΟΠΟΥ INPUT=INPUT FILE 41CHARES ΠΟΥ \*/  2 REC\_TYPE CHAR(02) INIT(' '),  2 I\_IP CHAR(11) INIT(' '),  2 BRANCH CHAR(05) INIT(' '),  2 FILLER01 CHAR(23) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | DCL 1 RHKS94PO BASED(ADDR(RKS94PO)), /\*\_HEADER RECORD\_45 BYTES\*/  2 REC\_TYPE CHAR(02) INIT(' '),  2 ABOUT\_TEXT CHAR(20) INIT(' '),  2 TIMESTAMP CHAR(19) INIT(' '),  2 FILLER01 CHAR(04) INIT(' ');  DCL 1 RTKS94PO BASED(ADDR(RKS94PO)), /\*\_TRAILER RECORD\_45 BYTES\*/  2 REC\_TYPE CHAR(02) INIT(' '),  2 COUNTER CHAR(07) INIT(' '),  2 FILLER02 CHAR(36) INIT(' ');  DCL 1 RDKS94PO BASED(ADDR(RKS94PO)),  /\*\_\_DETAIL RECORD ΑΝΑΛΥΤΙΚΕΣ ΕΓΓΡΑΦΕΣ \*/  /\*ΟΠΟΥ INPUT=INPUT FILE 45CHARES ΠΟΥ \*/  2 REC\_TYPE CHAR(02) INIT(' '),  2 I\_IP CHAR(11) INIT(' '),  2 BRANCH CHAR(05) INIT(' '),  2 FILLER03 CHAR(01) INIT(' '),  2 HOLDER CHAR(26) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | @SRVIIPS : CHECK IF SYDIPEL is individual Person ,PERSON\_TYPE = 'ΦΠ'  @SRVINMF : SEARCH THE FIRST,LAST,FATHER'S NAME |
| **Main Process Logic** | * Process to retrieve Current Dates GET\_CURRENT\_DATE * Open Input File 1 KS94PA * Check of Correct header ABOUT\_TEXT = 'MAS PREPAID KA' of Input File 1 KS94PA * Reading in a loop of Input File 1 KS94PA to get DETAIL RECORDS FD and count them in order to validate them with the number in FT trailer record. * Closing Input File 1 KS94PA * Open Input File 2 KS94PB , Outfile KS94PO * Read file KS94PB * Write header of KS94PB to Outfile KS94PO * Loop reading of KS94PB as described in the next Tab * Close Files KS94PB , KS94PO * Statistics of read and inserted input lines |
| **PROGRAM MAIN LOOP** | * For each DETAIL RECORDS FD of input file KS94PB do the following   + Increase counter WS\_INSERTED\_P   + Call routine CHECK\_SYDIPEL by calling service @SRVIIPS to validate that (@SRVIIPS.C\_IP\_TP = 'ΦΠ') entry is individual person   + Call routine CHECK\_NAMES by calling @SRVINMF to get SNAME ,FNAME , PNAME to get SURNAME, FIRSTNAME, MIDDLENAME for the output file.   + Write to output file   + Write footer |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

#### 

#### [KRPPD021] Massive Issuance KA Response Processing

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| KRPPD021 | KS75P | CRDP.PD.PRKEA.FTIP.CRD02101.SEQ  CRDP.PD.PRKEA.FTIP.CRD02101.SEQ | CRDP.PD.PRKEA.INTT.CRD02101.SEQ | RESPONSE from W4 (ISSUING VISA PREPAID KA) |
|  | BACK01 | CRDP.PD.PRKEA.FTIP.CRD02101.SEQ | KATP.PD.PRKEA.GDGT.CRD02101.BACK | BACKUP THE FTIP FILES BEFORE CLEARING THEM |
|  | CLRSEQ1 | CRDP.PD.PRKEA.FTIP.CRD02101.SEQ |  | CLEAR: VISA PREPAID KA: MASSIVE ISSUANCE FROM WAY4 |

#### [KS75P] Massive Issuance KA Program

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | DCL 1 RHKS75PI BASED(ADDR(RKS75PI)), /\*\_HEADER RECORD\_56 BYTES\*/  2 REC\_TYPE CHAR(02) INIT(' '),  2 BLANK01 CHAR(01) INIT(' '),  2 ABOUT\_TEXT CHAR(23) INIT(' '),  2 BLANK02 CHAR(01) INIT(' '),  2 TIMESTAMP CHAR(19) INIT(' '),  2 BLANK03 CHAR(10) INIT(' ');  DCL 1 RTKS75PI BASED(ADDR(RKS75PI)), /\*\_TRAILER RECORD\_56 BYTES\*/  2 REC\_TYPE CHAR(02) INIT(' '),  2 REC\_COUNT CHAR(07) INIT(' '),  2 BLANK04 CHAR(47) INIT(' ');  DCL 1 RDKS75PI BASED(ADDR(RKS75PI)),  /\*\_\_DETAIL RECORD ΑΝΑΛΥΤΙΚΕΣ ΕΓΓΡΑΦΕΣ \*/  /\*ΟΠΟΥ INPUT=INPUT FILE 56CHARES ΠΟΥ \*/  2 REC\_TYPE CHAR(02) INIT(' '),  2 I\_IP CHAR(11) INIT(' '),  2 BLANK05 CHAR(01) INIT(' '),  2 BRANCH CHAR(05) INIT(' '),  /\* FLAG\_NEW = '1' -> OLD  FLAG\_NEW = '0' -> NEW \*/  2 FLAG\_NEW CHAR(01) INIT(' '),  2 BLANK06 CHAR(01) INIT(' '),  2 CARD\_ID CHAR(16) INIT(' '),  2 BLANK07 CHAR(01) INIT(' '),  2 CARD CHAR(16) INIT(' '),  2 BLANK08 CHAR(01) INIT(' '),  2 FLAG CHAR(01) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | DCL 1 RHKS75PO BASED(ADDR(RKS75PO)), /\*\_HEADER RECORD\_57 BYTES\*/  2 REC\_TYPE CHAR(02) INIT(' '),  2 ABOUT\_TEXT CHAR(23) INIT(' '),  2 TIMESTAMP CHAR(19) INIT(' '),  2 FILLER01 CHAR(13) INIT(' ');  DCL 1 RTKS75PO BASED(ADDR(RKS75PO)), /\*\_TRAILER RECORD\_57 BYTES\*/  2 REC\_TYPE CHAR(02) INIT(' '),  2 COUNTER CHAR(07) INIT(' '),  2 FILLER02 CHAR(48) INIT(' ');  DCL 1 RDKS75PO BASED(ADDR(RKS75PO)),  /\*\_\_DETAIL RECORD ΑΝΑΛΥΤΙΚΕΣ ΕΓΓΡΑΦΕΣ \*/  /\*ΟΠΟΥ INPUT=INPUT FILE 57CHARES ΠΟΥ \*/  2 REC\_TYPE CHAR(02) INIT(' '),  2 I\_IP CHAR(11) INIT(' '),  2 BRANCH CHAR(05) INIT(' '),  2 CARD\_ID CHAR(20) INIT(' '),  2 CARD CHAR(19) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | @SRVIXAI: CREATE NEW CARD - CLIENT RELATION IN SYDIPEL  @ICCSPN:INSERT TO SMARTPIN TABLE  @ICDSPT: INSERT IN TABLE CRDT.SMARTPIN\_TRACK |
| **Main Process Logic** | * Open Input File 1 KS75PA * Check of Correct header ABOUT\_TEXT ='MASS ISS PREPAID\_KA\_RSP' of Input File 1 KS75PA * Reading in a loop of Input File 1 KS75PA to get DETAIL RECORDS FD and count them in order to validate them with the number in FT trailer record. * Closing Input File 1 KS75PA * Open Input File 2 KS75PB , Outfile KS75PO * Read file KS75PB * Write header of KS75PB to Outfile KS75PO * Loop reading of KS94PB as described in the next Tab * Close Files KS75PB, KS75PO * Statistics of read and inserted input lines |
| **PROGRAM MAIN LOOP** | * For each DETAIL RECORDS FD of input file KS94PB do the following   + Increase counter WS\_READ   + Check Input file RDKS75PI.FLAG     - when ‘1’ write output and increase WS\_SKIP counter     - When empty ‘ ’ check inputs RDKS75PI.FLAG\_NEW   For the case of FLAG\_NEW = '0' do the following   * + - Increase counter WS\_INSERTED\_P     - CALL CONNECT\_SYDIPEL, using RDKS75PI.I\_IP & RDKS75PI.CARD     - CALL INSERT\_SMARTPIN, using RDKS75PI.CARD     - CALL INSERT\_TRACK, using RDKS75PI.CARD     - WRITE OUTPUT     - CALL COMMIT\_RTN;     - Assign input flag to 1 RDKS75PI.FLAG = '1' increase counter WS\_FLAG\_NEW   For the case of FLAG\_NEW = '1' do the following   * + - WRITE OUTPUT     - Increase counter WS\_FLAG\_OLD      * + When footer Write footer to output   + Rewrite input file KS75PB FROM(RKS75PI) in order to depict the new value RDKS75PI.FLAG = '1'.   + Read next line |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.SMARTPIN\_TRACK  KCCT.SMARTPIN |

## [Way4 Archiving]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Way4 Archiving |
| **Batch Flow Title** | KRDPD006 |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | Way4 |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   The Way4 Archiving process is being used for the archiving of the parametric card table CRDT.W4\_TRN into the CRDT.W4\_TRN\_HIST. |

### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KRDPD006] Way4 Archiving

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRDPD006** | UNLW41 |  | CRDP.PD.W4TRA.SEMT.CRD00601.SEQ | Unload and remove table:CRDT.W4\_TRN and reorganize table space using DB2 Procedure DSNUPROC |
|  | POINT1 |  |  | Create a consistency point for tablespace CRDD001.CRDSWTRN using DB2 Procedure DSNUPROC |
|  | DELAY1 |  |  | Add delay of '00:02:00' |
|  | K30CS | CRDP.PD.W4TRA.SEMT.CRD00601.SEQ | CRDP.PD.W4TRA.SEMT.CRD00602.SEQ | Sort exported file CRDP.PD.W4TRA.SEMT.CRD00601.SEQ (table CRDT.W4\_TRN) for performance reasons |
|  | LOADWTRH | CRDP.PD.W4TRA.SEMT.CRD00602.SEQ |  | Load to table CRDT.W4\_TRN\_HIST using DB2 Procedure DSNUPROC |
|  | BACK | KATP.TMSTMP.END.SEQ | CRDP.PD.TMSTM.GDGT.CRD00608.BACK(+1) | Create a copy of the Input file to Output using utility IDCAMS and Repro function |
|  | BACKS | CRDP.PD.TMSTM.GDGT.CRD00608.BACK(-2) | CRDP.PD.ARCTM.SEMT.CRD00603.SEQ | Sort IN to OUT |
|  | UNLMU1 |  | CRDP.PD.MSGUN.SEMT.CRD00604.SEQ | Unload and remove table:CRDT.MSG\_UNIQ and reorganize table space CRDD001.CRDSMSUN using DB2 Procedure DSNUPROC |
|  | POINT2 |  |  | Create a consistency point for tablespace CRDD001.CRDSMSUN using DB2 Procedure DSNUPROC |
|  | DELAY2 |  |  | Add delay of '00:02:00' |
|  | K38CS | CRDP.PD.MSGUN.SEMT.CRD00604.SEQ | CRDP.PD.MSGUN.SEMT.CRD00605.SEQ | Sort exported file CRDP.PD.MSGUN.SEMT.CRD00605.SEQ (table CRDT.MSG\_UNIQ) for performance reasons |
|  | K38CP | CRDP.PD.MSGUN.SEMT.CRD00605.SEQ |  | Execute Loading program K38CP to load into table CRDT.MSG\_UNIQ\_ARC |
|  | UNMUH1 |  | CRDP.PD.MSGUH.SEMT.CRD00606.SEQ | Unload and remove table:CRDT.MSG\_UNIQ\_HIST and reorganize table space CRDD001.CRDSMSUHusing DB2 Procedure DSNUPROC |
|  | POINT3 |  |  | Create a consistency point for tablespace CRDD001.CRDSMSUH using DB2 Procedure DSNUPROC |
|  | K39CS | CRDP.PD.MSGUH.SEMT.CRD00606.SEQ | CRDP.PD.MSGUH.SEMT.CRD00607.SEQ | Sort exported file CRDP.PD.MSGUH.SEMT.CRD00606.SEQ (table CRDT.MSG\_UNIQ\_HIST) for performance reasons |
|  | K39CP | CRDP.PD.MSGUH.SEMT.CRD00607.SEQ |  | Execute Loading program K39CP to load into table CRDT.MSG\_UNIQ\_HIST\_ARC |
|  | UNMSE1 |  | CRDP.PD.MSUSE.SEMT.CRD00608.SEQ | Unload and remove table:CRDT.MSG\_UNIQ\_SESP and reorganize table space CRDD001.CRDSMUSE using DB2 Procedure DSNUPROC |
|  | POINT4 |  |  | Create a consistency point for tablespace CRDD001.CRDSMUSE using DB2 Procedure DSNUPROC |
|  | K88CS | CRDP.PD.MSUSE.SEMT.CRD00608.SEQ | CRDP.PD.MSUSE.SEMT.CRD00609.SEQ | Sort exported file CRDP.PD.MSUSE.SEMT.CRD00608.SEQ (table CRDT.MSG\_UNIQ\_SESPT) for performance reasons |
|  | K88CP | CRDP.PD.MSUSE.SEMT.CRD00609.SEQ |  | Execute Loading program K88CP to load into table CRDT.MSG\_UNIQ\_SESP\_ARC |
|  | UNMSH1 |  | CRDP.PD.MSGSH.SEMT.CRD00609.SEQ | Unload and remove table:CRDT.MSG\_UNIQ\_SESP\_HIST and reorganize table space CRDD001.CRDSMUSH using DB2 Procedure DSNUPROC |
|  | POINT5 |  |  | Create a consistency point for tablespace CRDD001.CRDSMUSH using DB2 Procedure DSNUPROC |
|  | K89CS | CRDP.PD.MSGSH.SEMT.CRD00609.SEQ | CRDP.PD.MSGSH.SEMT.CRD00610.SEQ | Sort exported file CRDP.PD.MSGSH.SEMT.CRD00609.SEQ (table CRDT.MSG\_UNIQ\_SESP\_HIST) for performance reasons |
|  | K89CP | CRDP.PD.MSGSH.SEMT.CRD00610.SEQ |  | Execute Loading program K89CP to load into table CRDT.MSG\_UNIQ\_SESP\_HIST\_ARC |
|  | DEALLOC | CRDP.PD.W4TRA.SEMT.CRD00601.SEQ  CRDP.PD.MSUSE.SEMT.CRD00609.SEQ  CRDP.PD.MSGSH.SEMT.CRD00610.SEQ  CRDP.PD.W4TRA.SEMT.CRD00602.SEQ  CRDP.PD.ARCTM.SEMT.CRD00603.SEQ  CRDP.PD.MSGUN.SEMT.CRD00604.SEQ  CRDP.PD.MSGUN.SEMT.CRD00605.SEQ  CRDP.PD.MSGUH.SEMT.CRD00606.SEQ  CRDP.PD.MSGUH.SEMT.CRD00607.SEQ  CRDP.PD.MSUSE.SEMT.CRD00608.SEQ  CRDP.PD.MSGSH.SEMT.CRD00609.SEQ |  | Delete all files used in the above processing steps using IBM Utility IEFBR14 |

#### [K38CP] Way4 Archiving - Loads table: CRDT.MSG\_UNIQ\_ARC from exported table:CRDT.MSG\_UNIQ

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | DCL 1 I\_REC,  5 SRN CHAR (012) INIT(' '),  5 TRDATE CHAR (010) INIT(' '),  5 MSG\_TYPE CHAR (004) INIT(' '),  5 TRN\_CDE CHAR (003) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | Reads sequentially input file : CRDP.PD.MSGUN.SEMT.CRD00605.SEQ and INSERTS into table: CRDT.MSG\_UNIQ\_ARC |
| **PROGRAM MAIN LOOP** | Main steps are:   * OPEN INPUT FILE * READ 1ST RECORD * IN A LOOP TILL THE EOF   + PERFORM INSERTS TO TABLE : CRDT.MSG\_UNIQ\_ARC   + EVERY 1000 INSERTIONS COMMIT WORK * AT END OF INPUT   + COMMIT WORK   + CLOSE INPUT FILE   + PRINT STATISTICS |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | * + CRDT.MSG\_UNIQ\_ARC |

#### [K39CP] Way4 Archiving - - Loads table: CRDT.MSG\_UNIQ\_SESP\_ARC from exported table:CRDT.MSG\_UNIQ\_SESP

#### 

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | DCL 1 I\_REC,  5 SRN CHAR (012) INIT(' '),  5 TRDATE CHAR (010) INIT(' '),  5 MSG\_TYPE CHAR (004) INIT(' '),  5 TRN\_CDE CHAR (003) INIT(' '),  5 UPDATE\_TIMESTAMP CHAR (026) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | Reads sequentially input file : CRDP.PD.MSUSE.SEMT.CRD00609.SEQ and INSERTS into table: CRDT.MSG\_UNIQ\_SESP\_ARC |
| **PROGRAM MAIN LOOP** | Main steps are:   * OPEN INPUT FILE * READ 1ST RECORD * IN A LOOP TILL THE EOF   + PERFORM INSERTS TO TABLE : CRDT.MSG\_UNIQ\_HIST\_ARC   + EVERY 1000 INSERTIONS COMMIT WORK * AT END OF INPUT   + COMMIT WORK   + CLOSE INPUT FILE   + PRINT STATISTICS |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | * + CRDT.MSG\_UNIQ\_ARC |

#### [K88CP] Way4 Archiving - Loads table: CRDT.MSG\_UNIQ\_SESP\_ARC from exported table:CRDT.MSG\_UNIQ\_SESP

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | DCL 1 I\_REC,  5 OR\_TYP CHAR(004) INIT(' '),  5 OR\_STAN CHAR(006) INIT(' '),  5 OR\_TRDATE CHAR(010) INIT(' '),  5 OR\_ACQID CHAR(006) INIT(' '),  5 RRN CHAR(012) INIT(' '),  5 ACQID CHAR(006) INIT(' '),  5 CARD CHAR(010) INIT(' '),  5 TYP CHAR(004) INIT(' '),  5 STAN CHAR(006) INIT(' '),  5 AUTHCODE CHAR(006) INIT(' '),  5 INSERT\_TIMESTAMP CHAR(026) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | Reads sequentially input file : CRDP.PD.MSUSE.SEMT.CRD00609.SEQ and INSERTS into table: CRDT.MSG\_UNIQ\_SESP\_ARC |
| **PROGRAM MAIN LOOP** | Main steps are:   * OPEN INPUT FILE * READ 1ST RECORD * IN A LOOP TILL THE EOF   + PERFORM INSERTS TO TABLE : CRDT.MSG\_UNIQ\_SESP\_ARC   + BYPASS DUPLICATE ROWS ERROR (-803)   + EVERY 1000 INSERTIONS COMMIT WORK * AT END OF INPUT   + COMMIT WORK   + CLOSE INPUT FILE   + PRINT STATISTICS |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | * + CRDT.MSG\_UNIQ\_ARC |

#### [K89CP] Way4 Archiving - Loads table: CRDT.MSG\_UNIQ\_SESP\_HIST\_ARC from exported table:CRDT.MSG\_UNIQ\_SESP\_HIST

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | DCL 1 I\_REC,  5 OR\_TYP CHAR(004) INIT(' '),  5 OR\_STAN CHAR(006) INIT(' '),  5 OR\_TRDATE CHAR(010) INIT(' '),  5 OR\_ACQID CHAR(006) INIT(' '),  5 RRN CHAR(012) INIT(' '),  5 ACQID CHAR(006) INIT(' '),  5 CARD CHAR(010) INIT(' '),  5 TYP CHAR(004) INIT(' '),  5 STAN CHAR(006) INIT(' '),  5 AUTHCODE CHAR(006) INIT(' '),  5 UPDATE\_TIMESTAMP CHAR(026) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | Reads sequentially input file : CRDP.PD.MSGSH.SEMT.CRD00610.SEQ and INSERTS into table: CRDT.MSG\_UNIQ\_SESP\_HIST\_ARC |
| **PROGRAM MAIN LOOP** | Main steps are:   * OPEN INPUT FILE * READ 1ST RECORD * IN A LOOP TILL THE EOF   + PERFORM INSERTS TO TABLE : CRDT.MSG\_UNIQ\_SESP\_HIST\_ARC   + BYPASS DUPLICATE ROWS ERROR (-803)   + EVERY 1000 INSERTIONS COMMIT WORK * AT END OF INPUT   + COMMIT WORK   + CLOSE INPUT FILE   + PRINT STATISTICS |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.MSG\_UNIQ\_SESP\_HIST\_ARC |

## [Recarding Prepaid Visa to MC]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Recarding Prepaid Visa to MC |
| **Batch Flow Title** | KRPPD039 |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | Way4 |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   The Recarding Prepaid Visa to MC process is being used to prepare the new recarded Prepaid MC Cards for the embossing process. More specifically, it inserts data to three cards table and one clients table. |

### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KRPPD039] Recarding Prepaid Visa to MC

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD039** | ALLOCAT |  | CRDP.PD.RCRDN.SEMT.CRD03902.SEQ  CRDP.PD.RCRDN.SEMT.CRD03903.SEQ | ALLOCATE OUTPUT FILES USING IBM UTILITY IEFBR14 |
|  | K92DP | CRDP.PD.RCRDN.FTIP.CRD03901.SEQ | CRDP.PD.RCRDN.SEMT.CRD03902.SEQ  CRDP.PD.RCRDN.SEMT.CRD03903.SEQ | READS INPUT FILE AND PERFORMS THE NECESSARY INSERTIONS TO THE TABLES TO ALLOW RECARDING OF PREPAID VISA TO MASTERCARD |
|  | STEPISE1 | CRDP.PD.RCRDN.SEMT.CRD03902.SEQ |  | COUNTS NUMBER OF RECORDS IN REJECTIONS INPUT FILE AND IF THERE ARE RECORDS PERFORM THE FOLLOWING STEPS |
|  | EMB1 |  | CRDP.PD.RCRDN.SEMT.K92MAIL1.SEQ | CREATE USING IDCAMS-REPRO THE OUPUT FILE WITH INFRORMATORY MESSAGE |
|  | EMAIL1 | CRDP.PD.RCRDN.SEMT.K92MAIL1.SEQ CRDP.PD.RCRDN.SEMT.CRD03902.SEQ | [NBGACSG@NBG.GR](mailto:NBGACSG@NBG.GR)  [CMS\_SUPPORT@NBG.GR](mailto:CMS_SUPPORT@NBG.GR) | TRANSMIT INFORMATORY FILE AND FILE WITH REJECTIONS TO EMAIL RECIPIENTS |
|  | DEALLOC1 | CRDP.PD.RCRDN.SEMT.K92MAIL1.SEQ |  | DELETE FILE USING IBM UTILITY IEFBR14 |
|  | STEPISE2 | CRDP.PD.RCRDN.SEMT.CRD03903.SEQ |  | CHECK IF SYDIPEL FILE IS EMPTY AND IF SO |
|  | EMB2 |  | CRDP.PD.RCRDN.SEMT.K92MAIL2.SEQ | PREPARE INFORMATORY FILE WITH SYDIPEL REJECTIONS USING IDCAMS-REPRO METHOD |
|  | EMAIL2 | CRDP.PD.RCRDN.SEMT.K92MAIL2.SEQ  CRDP.PD.RCRDN.SEMT.CRD03903.SEQ | [NBGACSG@NBG.GR](mailto:NBGACSG@NBG.GR)  [CMS\_SUPPORT@NBG.GR](mailto:CMS_SUPPORT@NBG.GR) | TRANSMIT INFORMATORY FILE AND FILE WITH SYDIPEL REJECTIONS TO EMAIL RECIPIENTS |
|  | DEALLOC2 | CRDP.PD.RCRDN.SEMT.K92MAIL2.SEQ |  | DELETE FILE USING IBM UTILITY IEFBR14 |
|  | DEALLOC3 | CRDP.PD.RCRDN.SEMT.CRD03902.SEQ  CRDP.PD.RCRDN.SEMT.CRD03903.SEQ |  | DELETE FILE USING IBM UTILITY IEFBR14 |

#### [K92DP] Recarding Prepaid Visa to MC -

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | DCL RK92DPI CHAR(106) INIT(' ') ;  DCL 1 RHK92DPI BASED(ADDR(RK92DPI)), /\*\_HEADER RECORD\_90 BYTES\*/  2 REC\_TYPE CHAR(02) INIT(' '),  2 TIMESTAMP CHAR(26) INIT(' '),  2 ABOUT\_TXT CHAR(18) INIT(' '),  2 BLANK01 CHAR(43) INIT(' ');  DCL 1 RTK92DPI BASED(ADDR(RK92DPI)), /\*\_TRAILER RECORD\_90 BYTES\*/  2 REC\_TYPE CHAR(02) INIT(' '),  2 REC\_COUNT CHAR(07) INIT(' '),  2 BLANK02 CHAR(80) INIT(' ');  DCL 1 RDK92DPI BASED(ADDR(RK92DPI)),  /\*\_\_DETAIL RECORD ΑΝΑΛΥΤΙΚΕΣ ΕΓΓΡΑΦΕΣ \*/  /\*ΟΠΟΥ INPUT=INPUT FILE 90CHARES ΠΟΥ \*/  2 REC\_TYPE CHAR(02) INIT(' '),  2 I\_IP CHAR(11) INIT(' '),  2 BLANK03 CHAR(01) INIT(' '),  2 CARD\_OLD CHAR(16) INIT(' '),  2 BLANK04 CHAR(01) INIT(' '),  2 CARD\_NEW CHAR(16) INIT(' '),  2 BLANK05 CHAR(01) INIT(' '),  2 BARCODE CHAR(20) INIT(' '),  2 BLANK06 CHAR(01) INIT(' '),  2 PROD\_CODE CHAR(03) INIT(' '),  2 BLANK07 CHAR(01) INIT(' '),  2 RESPONSE\_CD CHAR(02) INIT(' '),  2 RESPONSE\_TXT CHAR(30) INIT(' '),  2 FLAG\_CHECK CHAR(01) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | DCL 1 RK92DP2O,  2 I\_IP CHAR(11) INIT(' '),  2 FILLER01 CHAR(01) INIT(' '),  2 CARD\_OLD CHAR(16) INIT(' '),  2 FILLER02 CHAR(01) INIT(' '),  2 CARD\_NEW CHAR(16) INIT(' '),  2 FILLER03 CHAR(01) INIT(' '),  2 RESPONSE\_TXT CHAR(30) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | DCL 1 RK92DP3O,  2 I\_IP CHAR(11) INIT(' '),  2 FILLER01 CHAR(01) INIT(' '),  2 CARD\_OLD CHAR(16) INIT(' '),  2 FILLER02 CHAR(01) INIT(' '),  2 CARD\_NEW CHAR(16) INIT(' '),  2 FILLER03 CHAR(01) INIT(' '),  2 RESPONSE\_TXT CHAR(30) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | ERROR\_FOUND, READ\_K92DPI |
| **SERVICES** | SRVIXAI: CREATE NEW CARD - CLIENT RELATION IN SYDIPEL  ICCSPN: INSERT TO SMARTPIN TABLE  ICDSPT:INSERT IN TABLE CRDT.SMARTPIN\_TRACK  ICDERC:INSERT IN CRDT.EMBOS\_RECARDING  UCDERC:UPDATE CRDT.EMBOS\_RECARDING  SCDERC2:SELECT FROM CRDT.EMBOS\_RECARDING |
| **Main Process Logic** | MAIN PROCESS OF RECARDING VISA PREPAID CARD TO MASTERCARDS. PERFORMS ALL NECESSARY INSERTIONS TO DB2 TABLES THAT ACCOMPLISH THIS TASK |
| **PROGRAM MAIN LOOP** | THE MAIN STEPS ARE:   * OPEN AND VALIDATE INPUT FILE AS TO ITS HEADER/DETAIL & FOOTER COUNTERS, CLOSE FILE * OPEN INPUT AND OUTPUT FILES & READ 1ST RECORD * IN A LOOP TILL THE END OF THE INPUT FILE PERFORM THE FOLLOWING TASKS CONCERNING THE RECORD TYPE ‘FD’ AND RDK92DPI.FLAG\_CHECK IS SPACE:   + INVOKE SERVICE ICDERC TO INSERT INTO CRDT.EMBOS\_RECARDING   + INVOKE SERVICE SRVIXAI TO ESTABLISH SYDIPEL CONNECTION, using RDK92DPI.I\_IP & RDK92DPI.CARD\_NEW   + INVOKE SERVICE SMARTPIN TO INSERT INTO TABLE KCCT.SMARTPIN, using RDK92DPI.CARD\_NEW   + INVOKE SERVICE ICDSPT TO INSERT INTO CRDT.SMARTPIN\_TRACK, using RDK92DPI.CARD\_NEW   + INVOKE SERVICE CARDSPIN TO INSERT INTO KCCT.CARDS\_PIN, using RDK92DPI.CARD\_NEW & RDK92DPI.BARCODE   + UPDATE INPUT FILE WITH RDK92DPI.FLAG\_CHECK = '1' SO THAT, IN CASE OF A RERUN TO BYPASS THIS RECORD AS PROCESSED * ALL THE ABOVE TASKS ARE CARRIED OUT WHEN THERE ARE NO ERROR CODES RETURNED FROM THE DIFFERENT SERVICE INVOCATIONS, OTHERWISE THE INPUT RECORD IS WRITTEN TO THE CMOD FILE AS REJECTED |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_RECARDING  KCCT.SMARTPIN  CRDT.SMARTPIN\_TRACK  KCCT.CARDS\_PIN |

## [Merge Sydipel]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Merge Sydipel |
| **Batch Flow Title** | KRDPD021 |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | Way4 |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   Merge Sydipel process retrieves details of client's remaining client IDS for daily merge of client IDS operations. |

### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KRDPD021] Merge Sydipel

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| KRDPD021 | K23CS1 | DFRP.FTOEAEDS.SEQ | CRDP.PD.MERSY.SEMT.CRD02101.SEQ | Sorting by REFORMAT DFRP.FTOEAEDS.SEQ FILE - KEEP ONLY THE COLUMNS:  REPLACED CLIENTS SYDIPEL , REMAINING CLIENTS SYDIPEL  ,ALSO ADD LEADING CHAR(1) FIELD TO EACH SYDIPEL NO |
|  | K23CS2 | CRDP.PD.MERBR.PERP.KAMSYN01.SEQ | CRDP.PD.MERSY.SEMT.CRD02102.SEQ | REMOVE HEADER AND TRAILER FROM KAMSYN OUTPUT FILE |
|  | K23CS3 | CRDP.PD.MERSY.SEMT.CRD02101.SEQ  CRDP.PD.MERSY.SEMT.CRD02102.SEQ | CRDP.PD.MERSY.SEMT.CRD02103.SEQ | MERGE KAMSYN OUTPUT FILE AND DFRP.FTOEAEDS.SEQ REFORMATED FILE  (IN THIS STEP BOTH FILES HAVE THE SAME COLUMNS AND RECORD LENGTH) |
|  | K23CP | CRDP.PD.MERSY.SEMT.CRD02103.SEQ | CRDP.PD.MERSY.SEMT.CRD02104.SEQ | EXECUTEK23CP PROGRAM for TYPE OF CUSTOMER INDICATION , INDICATION IF CUSTOMER IS STAFF |
|  | K23CS4 | CRDP.PD.MERSY.SEMT.CRD02104.SEQ | CRDP.PD.MERSY.FTOP.CRD02106.SEQ | INSERT FILE HEADER AND FILE TRAILER  AND COPY FINAL FILE TO PERMANENT FTP FILE |
|  | CLEAR01 |  | CRDP.PD.MERBR.PERP.KAMSYN01.SEQ | CLEAR PERP FILE CONTENTS |
|  | DEALLOC |  | CRDP.PD.MERSY.SEMT.CRD02101.SEQ  CRDP.PD.MERSY.SEMT.CRD02102.SEQ  CRDP.PD.MERSY.SEMT.CRD02103.SEQ  CRDP.PD.MERSY.SEMT.CRD02104.SEQ | DELETE TEMPORARY FILES |

#### 

#### [K23CP] Merge Sydipel

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | DCL 1 I\_REC,  5 REPLACED\_CUST CHAR (011) INIT(' '),  5 REMAINING\_CUST CHAR (011) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | DCL 1 O\_REC,  5 REPLACED\_CUST CHAR (011) INIT(' '),  5 REMAINING\_CUST CHAR (011) INIT(' '),  5 TYPE\_IND\_RMN CHAR (004) INIT(' '),  5 STAFF\_IND\_RMN CHAR (008) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | @SRVIIPS : Call to validate type of Customer  @SRVCKSF : Call to validate staff Data |
| **Main Process Logic** | Program K23CP performs in sequence the following processes  ● Opens Input and Output files  ● Reads the first record of input File  ● Execute the loop until end of file as described in the below tab  ● Closes Input and Output files  ● Writes Statistics counter messages |
| **PROGRAM MAIN LOOP** | * Process all the input Lines record by record * For each record, through the procedure RETR\_CUST\_TP invokes service @SRVIIPS to get the type of customer L\_CUST\_TYPE = @SRVIIPS.ODATA.C\_IP\_TP. * On successful operation rc=0 when PHYSICAL OR LEGAL ENTITY (L\_CUST\_TYPE ^= 'MO' & (L\_CP)) it calls CHK\_IF\_STAFF procedure which invokes service @SRVCKSF that checks if customer is staff L\_STF = @SRVCKSF.ODATA.CLS\_STAFF. * Writes output , increase counter RECORDS\_WRITTEN\_OUT1 * Initialize variables used * Read next line , increase counter RECORDS\_READ1 |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

## 

## [Credit Statements]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Credit Statements |
| **Batch Flow Title** | KRCPD010 |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | Way4 |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   Credit Statments process is being used to print a credit card’s holder transactions. It uses client services to collect the aprropriate data regarding the name or the address of the credit card’s holder. |

### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KRCPD010] Credit Statements Processing

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| KRCPD010 | K63DP | CRDP.PD.PRSTM.FTIP.CRD01001.SEQ | CRDP.PD.PRSTM.FTOP.CRD01001.SEQ | READS FLAT FILE FROM IL AND CREATES FILE FOR CREDIT STATEMENTS FOR CMOD |

#### [K63DP] Credit Statements Program

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | DCL 1 RK63DPIN1, /\* RECORD LENGTH 95 \*/  2 ID CHAR(36),  2 SEP1 CHAR(01),  2 CUSTOMER\_ID PIC'(11)Z',  2 SEP2 CHAR(01),  2 BUSINESS\_ID PIC'(11)Z',  2 SEP3 CHAR(01),  2 PAN CHAR(16),  /\* START\_CODE U714 E.PAPPAS - 23.05.2022\*/  2 SEP4 CHAR(01),  2 ISSUING\_CONTRACT CHAR(17); |
| **OUTFILE1 FILE DECLARATION** | DCL 1 ROUTFILE1,  2 ID CHAR(36) INIT(' '),  2 SEP1 CHAR(01) INIT('|'),  2 CUSTOMER\_ID PIC'(11)Z' INIT(' '),  2 SEP2 CHAR(01) INIT('|'),  2 BUSINESS\_ID PIC'(11)Z' INIT(' '),  2 SEP3 CHAR(01) INIT('|'),  2 C\_TIN CHAR(16) INIT(' '),  2 SEP4 CHAR(01) INIT('|'),  2 C\_ADDRESS CHAR(80) INIT(' '),  2 SEP5 CHAR(01) INIT('|'),  2 C\_ZIPCODE CHAR(40) INIT(' '),  2 SEP6 CHAR(01) INIT('|'),  2 C\_CITY CHAR(80) INIT(' '),  2 SEP7 CHAR(01) INIT('|'),  2 C\_NAME CHAR(80) INIT(' '),  2 SEP8 CHAR(01) INIT('|'),  2 B\_TIN CHAR(16) INIT(' '),  2 SEP9 CHAR(01) INIT('|'),  2 B\_ADDRESS CHAR(80) INIT(' '),  2 SEP10 CHAR(01) INIT('|'),  2 B\_ZIPCODE CHAR(40) INIT(' '),  2 SEP11 CHAR(01) INIT('|'),  2 B\_CITY CHAR(80) INIT(' '),  2 SEP12 CHAR(01) INIT('|'),  2 B\_NAME CHAR(80) INIT(' '),  2 SEP13 CHAR(01) INIT('|'),  2 STMT\_MAIL CHAR(80) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | @SRVIIPS Call to validate type of Customer and retrieve VAT number  @SRVIADR call to retrieve addresses  @SRVINMF call to retrieve name surname or name of business from IMF Databases  @SPRADR call to retrieve emails  @SRVILOC call to retrieve other addresses of customer |
| **Main Process Logic** | * Retrieve Current Date by calling `GET\_CURRENT\_DATE\_RTN` to retrieve and set the current date. * Opens input and output files. * Calls `READ\_INPUT\_FILE1` to read the first record. * Main Loop of program described below * Closes the input and output files. |
| **PROGRAM MAIN LOOP** | * Clears and sets separators '|' for the output record. * Sets `ID`, `CUSTOMER\_ID`, and `BUSINESS\_ID` in the output record. * Initially calls GET\_CUSTOMER\_DATA , flagging all next service calls as individuals CUS and calling @SRVIIPS to check validity of Individual person and get VAT number of customer , then proceeds with routine SELECT\_NAMES\_RTN (checking @SRVIIPS.ODATA.C\_IP\_TP for individual or business and calling @SRVINMF to get SNAME ,FNAME , PNAME using class CLASS = 900001 to call @SRVINMF ) to get SURNAME, FIRSTNAME, MIDDLENAM for C\_NAME for the output file.Then proceeds with routine SELECT\_ADDRESS\_TYPE\_RTN , calling @SRVILOC to take the field of address classification and using it in service @SRVIADR with address class WS\_ADDR\_CLASS = 1500003 @SRVILOC.ADDR\_CLSF filling the ADDRESS , CITY ZIP\_CODE with variations depending the @SRVIADR.ODATA.I\_CLSF\_ADDR\_STRC\_K classification 2700001 , 2700002 ,2700003,2700005 * ‘Calls GET\_BUSINESS\_DATA’ flagging all next service calls as business BUS and calling @SRVIIPS to check validity of business and get VAT number of business customer , then proceeds with routine SELECT\_NAMES\_RTN (checking @SRVIIPS.ODATA.C\_IP\_TP for individual or business and calling @SRVINMF to get BUSINESS\_NAME1 ,BUSINESS\_NAME2 , BUSINESS\_NAME3 using class CLASS = 900003 to call @SRVINMF ) to get B\_NAME for the output file. Then proceeds with routine SELECT\_ADDRESS\_TYPE\_RTN , calling @SRVILOC to take the field of address classification and using it in service @SRVIADR with address class WS\_ADDR\_CLASS = @SRVILOC.ADDR\_CLSF when no address found filling the ADDRESS , CITY ZIP\_CODE with variations depending the @SRVIADR.ODATA.I\_CLSF\_ADDR\_STRC\_K classification 2700001 , 2700002 ,2700003,2700005 * Finally from `GET\_STMT\_MAIL\_DATA` calls @SPRADR to get email using specific card digits @SPRADR.IDATA.ACCOUNT = SUBSTR(ISSUING\_CONTRACT,5,13) .If not found assigns null. * WRITE\_OUTPUT\_FILE1` to write the processed record. * Calls `READ\_INPUT\_FILE1` to read the next record. |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | KCCT.PR\_ADDRS |

## [Instant Issuing Debit & Dual cards]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Instant Issuing Debit & Dual cards |
| **Batch Flow Title** | KRDPD009 & KRDPD003 |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | Way4 |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   Instant Issuing process for Debit & Dual cards is being used to create the specific amount of instant cards that a bank branch demands. It uses a parametric card table as well as client table for validations. |

### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KRDPD009] Request Instant Issuing INSTANT DEBIT MASTERCARD & ETHNODEPOSIT

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| KRDPD009 | K13CP | CRDT.MASS\_ISS\_REQ | CRDP.PD.DCORG.SEMT.CRD00901.SEQ  CRDP.PD.DCORG.SEMT.CRD00902.SEQ | RETRIEVES |
|  | K13CS1 | CRDP.PD.DCORG.SEMT.CRD00901.SEQ | CRDP.PD.DCORG.FTOP.CRD00902.SEQ | INSERTS FILE HEADER, FILE TRAILER |
|  | K13CS2 | CRDP.PD.DCORG.SEMT.CRD00902.SEQ | CRDP.PD.DCORG.FTOP.CRD00903.SEQ | INSERTS FILE HEADER, FILE TRAILER(DUAL) |
|  | BACKUP1 | CRDP.PD.DCORG.FTOP.CRD00902.SEQ  CRDP.PD.DCORG.FTOP.CRD00903.SEQ | CRDP.PD.DCORG.GDGT.CRD00902.BACK(+1) | BACKUPS FILES USING IDCAMS REPRO |
|  | DEALLOC | CRDP.PD.DCORG.SEMT.CRD00901.SEQ  CRDP.PD.DCORG.SEMT.CRD00902.SEQ |  | DEALLOCATION OF FILES USING IBM UTILITY IEFBR14 |

#### [KRDPD003] Response Instant Issuing

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| KRDPD003 | K10CP | CRDP.PD.DCORG.FTIP.CRD00301.SEQ | CRDT.DEB\_FILE\_EXC (UPDATES) | RECORD VALIDATIONS OF ETHNODEPOSIT ISSUANCE AND INSTANT |
|  | K10CP2 | CRDP.PD.DCORG.FTIP.CRD00302.SEQ | CRDT.DEB\_FILE\_EXC (UPDATES) | RECORD VALIDATIONS OF DUAL/COMBO CARD INSTANT ISSUANCE |
|  | K58CS1 | CRDP.PD.DCORG.FTIP.CRD00301.SEQ  CRDP.PD.DCORG.FTIP.CRD00302.SEQ | CRDP.PD.DCORG.INTT.CRD00310.SEQ | CREATE FILE FOR INSTANT CMOD USING SORT |
|  | K11CS1 | CRDP.PD.DCORG.FTIP.CRD00301.SEQ  CRDP.PD.DCORG.FTIP.CRD00302.SEQ | CRDP.PD.DCORG.SEMT.CRD00302.SEQ | MERGE ALL ISSUANCE DETAILS USING SORT |
|  | K11CP | CRDP.PD.DCORG.SEMT.CRD00302.SEQ | CRDP.PD.DCORG.SEMT.CRD00303.SEQ  CRDP.PD.DCORG.SEMT.CRD00304.SEQ | CARD/CUSTOMER CONNECTION IN SIDIPEL USING PROGRAM K11CP |
|  | ICE01 | CRDP.PD.DCORG.SEMT.CRD00304.SEQ | CRDP.PD.DCORG.SEMT.CRD00305.SEQ | COUNT NUMBER OF OCCURRENCES OF EACH KEY |
|  | K04CS | CRDP.PD.DCORG.SEMT.CRD00305.SEQ | CRDP.PD.DCORG.SEMT.CRD00306.SEQ | REFORMAT FILE FOR CRDT.MASS\_ISS\_REQ USING SORT |
|  | K04CP | CRDP.PD.DCORG.SEMT.CRD00306.SEQ | CRDT.MASS\_ISS\_REQ UPDATES | UPDATE CRDT\_MASS\_ISS\_REQ |
|  | K11CS2 |  | CRDP.PD.DCORG.FTIP.CRD00301.SEQ | INITIALIZE FILE INSTANT AND ETHNODEPOSIT USING SORT |
|  | K11CS3 |  | CRDP.PD.DCORG.FTIP.CRD00302.SEQ | INITIALIZE DUAL INSTANT FILE USING SORT |
|  | DEALLOC | CRDP.PD.DCORG.SEMT.CRD00302.SEQ  CRDP.PD.DCORG.SEMT.CRD00303.SEQ  CRDP.PD.DCORG.SEMT.CRD00304.SEQ  CRDP.PD.DCORG.SEMT.CRD00305.SEQ  CRDP.PD.DCORG.SEMT.CRD00306.SEQ |  | DEALLOCATION OF FILES USING IBM UTILITY IEFBR14 |

#### [K13CP] RETRIEVES REQUESTS FROM CRDT.MASS\_ISS\_REQ FOR ETHNODEPOSIT AND INSTANT DEBIT MASTERCARD ISSUANCE

| **IL Process** | |
| --- | --- |
| **Process Name** | K13CP |
| **INPFILE1 FILE DECLARATION** | TABLE: CRDT.MASS\_ISS\_REQ |
| **OUTFILE1 FILE DECLARATION** | DCL 1 OUTPUT\_DATA\_FILE,  5 O\_APPLICATION\_NUMBER CHAR (023) INIT(' '),  5 O\_APPL\_SEQ\_NUMBER CHAR (005) INIT(' '),  5 O\_APPLICATION\_STATUS CHAR (001) INIT(' '),  5 O\_PRODUCTION\_EVENT CHAR (003) INIT(' '),  5 O\_IP CHAR (011) INIT(' '),  5 O\_CLS\_CLIENT CHAR (002) INIT(' '),  5 O\_CLS\_STAFF CHAR (008) INIT(' '),  5 O\_ACCOUNT CHAR (016) INIT(' '),  5 O\_ACCOUNT\_TYPE CHAR (002) INIT(' '),  5 O\_ACCOUNT\_DESCRIPTION CHAR (010) INIT(' '),  5 O\_CARD\_TYPE CHAR (002) INIT(' '),  5 O\_PRODUCT\_CODE CHAR (003) INIT(' '),  5 O\_PRODUCT\_GROUP CHAR (002) INIT(' '),  5 O\_FIRST\_NAME CHAR (032) INIT(' '),  5 O\_LAST\_LAME CHAR (032) INIT(' '),  5 O\_COMPANY\_NAME CHAR (032) INIT(' '),  5 O\_ISSUING\_BRANCH CHAR (003) INIT(' '),  5 O\_PIN\_DELIVERY\_BRANCH CHAR (003) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | FCDMIR2.ODATA.ROWS |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | FCDMIR2:ESTABLISH CONNECTION IN SIDIPEL  SRVCKSF:RETRIEVE INFORMATION IF THE CLIENT IS STAFF OR NOT |
| **Main Process Logic** | RETRIEVE REQUESTS FROM CRDT.MASS\_ISS\_REQ FOR ETHNODEPOSIT AND INSTANT DEBIT MASTERCARD ISSUANCE |
| **PROGRAM MAIN LOOP** | PROGRAM INITIALLY SETS CURRENT DATE AND TIME AND AFTER OPENING THE OUTPUT FILES ENTERS A LOOP WHERE:   * ESTABLISHES CONNECTION WITH SIDIPEL (STATUS = 0) * FOR EVERY FETCHED ROW INVOKES SERVICE SRVCKSF TO CHECK IS STAFF OR NOT * FOR EVERY CARD OF THE CLIENT WRITES TO THE OUTPUT FILE (WHEN PROD\_CODE = 'C06' WRITES FOR DUAL TO OUTFILE2) * AT END (L\_ROWS\_MORE NOT = ‘Y’) CLOSES FILES AND TERMINATES |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.MASS\_ISS\_REQ |

## 

#### [K10CP] RECORD VALIDATIONS OF ETHNODEPOSIT ISSUANCE AND INSTANT

| **IL Process** | |
| --- | --- |
| **Process Name** |  |
| **INPFILE1 FILE DECLARATION** | DCL INPFILE1 FILE RECORD INPUT;  DCL 1 RECFIELD,  5 ROWCODE CHAR (002) INIT(' '),  5 REST\_DETAILS CHAR (317) INIT(' ');  DCL 1 FH\_INFL1 BASED (ADDR(RECFIELD)),  2 FH\_ROWCODE CHAR(02) INIT(' '),  2 FH\_TIMESTP CHAR(26) INIT(' '),  2 FILLER1 CHAR(291) INIT(' ');  DCL 1 FTRL\_INPFILE1 BASED(ADDR(RECFIELD)),  5 FTRL\_ROWCODE CHAR (002) INIT(' '),  5 FTRL\_TOTAL\_RECS PIC'(07)9' INIT(0),  5 FILLER2 CHAR (310) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | SCDDFE : SELECTS FROM CRDT.DEB\_FILE\_EXC  UCDDFE: UPDATE CRDT.DEB\_FILE\_EXC |
| **Main Process Logic** | READS INPUT FILE AND VALIDATES ITS STRUCTURE AND CHECKS MULTIRUN ON THE SAME DATE |
| **PROGRAM MAIN LOOP** | READS INPUT FILE AND VALIDATES ITS STRUCTURE AS TO THE EXISTENCE OF HEADER / DETAIL AND FOOTER RECORDS ALONG WITH CRITICAL COUNTER OF INPUT RECORDS COUNTED AND REFERRED TO THE TRAILER RECORD. IT ALSO CHECKS IF MULTIPLE RUNS ON THE SAME DATE ARE ALLOWED AND UPDATE THE CURRENT TIMESTAMP OF THE RUN |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.DEB\_FILE\_EXC |

## 

#### [K11CP] ESTABLISHES CONNECTION OF CARD AND CUSTOMER IN SIDIPEL

| **IL Process** | |
| --- | --- |
| **Process Name** |  |
| **INPFILE1 FILE DECLARATION** | INPFILE1:  DCL 1 RECFIELD,  5 ROWCODE CHAR (002) INIT(' '),  5 FILLER\_1 CHAR (125) INIT(' '),  5 APPL\_STATUS CHAR (001) INIT(' '),  5 FILLER\_2 CHAR (191) INIT(' ');  DCL 1 INPUT\_DATA\_FILE BASED(ADDR(RECFIELD)),  5 I\_ROWCODE CHAR (002) INIT(' '),  5 I\_PRODUCTION\_EVENT CHAR (003) INIT(' '),  5 I\_REG\_NUM CHAR (032) INIT(' '),  5 I\_LEGAL\_IP CHAR (011) INIT(' '),  5 I\_PHYSICAL\_IP CHAR (011) INIT(' '),  5 I\_OLD\_DEBIT\_CARD CHAR (019) INIT(' '),  5 I\_NEW\_DEBIT\_CARD CHAR (019) INIT(' '),  5 I\_OLD\_PRODUCT\_CODE CHAR (009) INIT(' '),  5 I\_NEW\_PRODUCT\_CODE CHAR (009) INIT(' '),  5 I\_PRODUCT\_GROUP CHAR (002) INIT(' '),  5 I\_OWNER\_BRANCH CHAR (005) INIT(' '),  5 I\_PIN\_DELIVERY\_BRANCH CHAR (005) INIT(' '),  5 I\_APPL\_STATUS CHAR (001) INIT(' '),  5 I\_FREE\_TEXT\_1 CHAR (080) INIT(' '),  5 I\_FREE\_TEXT\_2 CHAR (080) INIT(' '),  5 I\_PRODUCT\_NAME CHAR (030) INIT(' '),  5 I\_SMARTPIN\_IND CHAR (001) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | DCL 1 OUTPUT\_DATA\_FILE,  5 O\_PRODUCTION\_EVENT CHAR (003) INIT(' '),  5 O\_REG\_NUM CHAR (032) INIT(' '),  5 O\_PHYSICAL\_IP CHAR (011) INIT(' '),  5 O\_OLD\_DEBIT\_CARD CHAR (016) INIT(' '),  5 O\_NEW\_DEBIT\_CARD CHAR (016) INIT(' '),  5 O\_OLD\_PRODUCT\_CODE CHAR (009) INIT(' '),  5 O\_NEW\_PRODUCT\_CODE CHAR (009) INIT(' '),  5 O\_OWNER\_BRANCH CHAR (005) INIT(' '),  5 O\_PIN\_DELIVERY\_BRANCH CHAR (005) INIT(' '),  5 O\_APPL\_STATUS CHAR (001) INIT(' '),  5 O\_FREE\_TEXT\_1 CHAR (080) INIT(' '),  5 O\_FREE\_TEXT\_2 CHAR (080) INIT(' ');  DCL 1 OUTPUT\_DATA\_FILE\_2,  5 OS\_REG\_NUM CHAR (032) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | @SRVIXAI: CREATE NEW CARD - CLIENT RELATION IN SYDIPEL  @ICCSPN:INSERT TO SMARTPIN TABLE  @ICDSPT: INSERT IN TABLE CRDT.SMARTPIN\_TRACK |
| **Main Process Logic** | ESTABLISHES CONNECTION OF CARD AND CUSTOMER IN SIDIPEL |
| **PROGRAM MAIN LOOP** | FOR EVERY RECORD OF THE INPUT FILE, WHERE ITS A DETAIL RECORD AND STATUS = 0 (MEANING IT MUST BE UPDATED) AND BASED ON THE PRODUCT GROUP, WHEN :   * CONSUMER PRODUCT (‘01’)   + PREPARE DATA FOR FIRST CLIENT   + ESTABLISH CONNECTION IN SIDIPEL * BUSINESS PRODUCT(‘02’)   + PREPARE DATA FOR FIRST CLIENT   + ESTABLISH CONNECTION IN SIDIPEL   + ESTABLISH CONNECTION OF PHYSICAL ENTITY   + ESTABLISH CONNECTION FOR CARD * ETHNODEPOSIT(‘03’)   + ESTABLISH CONNECTION OF ENTITY/CARD   + ESTABLISH CONNECTION WITH INDIVIDUAL COMPANY * SELF-EMPLOYED PRODUCT(‘05’)   + PREPARE DATA FOR FIRST CLIENT   + ESTABLISH CONNECTION IN SIDIPEL   FOR ALL PRODUCTS EXCEPT ETHNODEPOSIT INSERT INTO:   * INVOKE SERVICE ICCSPN AND INSERT INTO KCCT.SMARTPIN * INVOKE SERVICE ICDSPT AND INSERT INTO CRDT.SMARTPIN\_TRACK   AT END OF INPUT FILE CLOSES OUTPUT, PRINTS STATISTICS AND RETURNS TO THE O/S |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | * KCCT.SMARTPIN * CRDT.SMARTPIN\_TRACK |

#### 

#### 

#### [K04CP] UPDATE CRDT.MASS\_ISS\_REQ

| **IL Process** | |
| --- | --- |
| **Process Name** |  |
| **INPFILE1 FILE DECLARATION** | DCL INPFILE1 FILE RECORD INPUT;  DCL 1 INPUT\_DATA\_1,  5 I\_REG\_NUM CHAR (020),  5 I\_COUNTER PIC'99999'; |
| **OUTFILE1 FILE DECLARATION** | UPDATE CRDT.MASS\_ISS\_REQ |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | **SCDMIR**:SELECTS FROM CRDT.MASS\_ISS\_REQ  **UCDMIR**:UPDATES CRDT.MASS\_ISS\_REQ |
| **Main Process Logic** | PROCESSES INPUT FILE CONTAINING CARDS (AS PREPARED BY PROGRAM K11CP ) AND UPDATES CRDT.MASS\_ISS\_REQ |
| **PROGRAM MAIN LOOP** | FOR EVERY CARD OF THE INPUT FILE:   * SELECT FROM CRDT.MASS\_ISS\_REQ * UPDATE TABLE CRDT.MASS\_ISS\_REQ |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.MASS\_ISS\_REQ |

## 

## [Massive Issuance Debit cards]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Massive Issuance Debit cards |
| **Batch Flow Title** | KRDPD013 & KRDPD029 |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | Way4 |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   This process is being used for DEBIT MASTERCARD CONSUMER and DEBIT MASTERCARD BUSINESS massive issuance case. It uses client and card tables for validations and to collect the appropriate data. It also delivers file containing details and parameterization of debit card to be created in W4. |

### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KRDPD013] MASS ISSUANCE REQUESTS FOR DEBIT MASTERCARD CONSUMER AND BUSINESSES

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| KRDPD013 | K12CP | KATP.PD.DEBMA.PERP.KFB3POUT.SEQ  CRDP.PD.MASIS.PERP.CRD01301.SEQ | CRDP.PD.MASIS.SEMT.CRD01302.SEQ | MASSIVE ISSUANCE OF CONSUMER AND BUSINESS CARDS |
|  | K12CS | CRDP.PD.MASIS.SEMT.CRD01302.SEQ | CRDP.PD.MASIS.FTOP.CRD01303.SEQ | INSERT FILE HEADER, FILE TRAILER |
|  | CTRNORST | CRDP.PD.MASIS.SEMT.CRD01302.SEQ |  | DEALLOCATION OF FILES USING IBM IEFBR14 UTILITY |

#### [KRDPD029] SIDIPEL CONNECTION FOR MASSIVE ISSUANCE (DEBIT MASTERCARD CONSUMER AND BUSINESS)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| KRDPD029 | K10CP | CRDP.PD.DCORG.FTIP.CRD02901.SEQ |  |  |
|  | K11CS | CRDP.PD.DCORG.FTIP.CRD02901.SEQ | CRDP.PD.DCORG.SEMT.CRD02902.SEQ | MERGE ALL ISSUANCE DETAILS USING SORT - EXPORT ONLY ‘FD’ RECORDS |
|  | K11CP | CRDP.PD.DCORG.SEMT.CRD02902.SEQ | CRDP.PD.DCORG.SEMT.CRD02903.SEQ  CRDP.PD.DCORG.SEMT.CRD02904.SEQ | CARD/CUSTOMER CONNECTION IN SIDIPEL |
|  | INIT01 |  | CRDP.PD.DCORG.FTIP.CRD02901.SEQ | REMOVE FILE HEADER AND FOOTER |
|  | DEALLOC | CRDP.PD.DCORG.SEMT.CRD02902.SEQ  CRDP.PD.DCORG.SEMT.CRD02903.SEQ  CRDP.PD.DCORG.SEMT.CRD02904.SEQ |  | DEALLOCATE - DELETE FILES USING IBM UTILITY IEFBR14 |

#### [K12CP] **MASSIVE ISSUANCE OF CONSUMER AND BUSINESS CARDS**

| **IL Process** | |
| --- | --- |
| **Process Name** |  |
| **INPFILE1 FILE DECLARATION** | DCL OUTFILE1 FILE RECORD OUTPUT;  DCL 1 OUTPUT\_DATA\_1,  5 O\_IPP CHAR(11) INIT(' '),  5 O\_TEXT CHAR(114) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | DCL OUTFILE2 FILE RECORD OUTPUT;  DCL 1 OUTPUT\_DATA\_2,  5 O\_LEGAL\_I\_IP CHAR(011) INIT(' '),  5 O\_CLS\_CLIENT\_LEGAL\_CODE CHAR(010) INIT(' '),  5 O\_CLS\_CLIENT\_LEGAL\_VALUE CHAR(002) INIT(' '),  5 O\_PHYSICAL\_I\_IP CHAR(011) INIT(' '),  5 O\_CLS\_CLIENT\_PHYSICAL\_CODE CHAR(010) INIT(' '),  5 O\_CLS\_CLIENT\_PHYSICAL\_VALUE CHAR(002) INIT(' '),  5 O\_CLS\_STAFF\_PHYSICAL\_CODE CHAR(009) INIT(' '),  5 O\_CLS\_STAFF\_PHYSICAL\_VALUE CHAR(008) INIT(' '),  5 O\_ACCOUNT CHAR(016) INIT(' '),  5 O\_ACCOUNT\_TYPE CHAR(002) INIT(' '),  5 O\_ACCOUNT\_DESCRIPTION CHAR(010) INIT(' '),  5 O\_FIRST\_NAME CHAR(032) INIT(' '),  5 O\_LAST\_NAME CHAR(032) INIT(' '),  5 O\_COMPANY\_NAME CHAR(032) INIT(' '),  5 O\_ISSUING\_BRANCH CHAR(005) INIT(' '),  5 O\_PRODUCT\_GROUP CHAR(002) INIT(' '),  5 O\_PRODUCT\_CODE CHAR(009) INIT(' '),  5 O\_CASH\_GREECE\_IS\_DFLT CHAR(001) INIT(' '),  5 O\_CASH\_GREECE\_LMT PIC'(12)9V.99' INIT(' '),  5 O\_CASH\_OTHER\_IS\_DFLT CHAR(001) INIT(' '),  5 O\_CASH\_OTHER\_LMT PIC'(12)9V.99' INIT(' '),  5 O\_CASH\_B24\_IS\_DFLT CHAR(001) INIT(' '),  5 O\_CASH\_B24\_LMT PIC'(12)9V.99' INIT(' '),  5 O\_RETAIL\_WITHOUT\_EC\_IS\_DFLT CHAR(001) INIT(' '),  5 O\_RETAIL\_WITHOUT\_EC\_LMT PIC'(12)9V.99' INIT(' '),  5 O\_CNP\_GREECE\_IS\_DFLT CHAR(001) INIT(' '),  5 O\_CNP\_GREECE\_LMT PIC'(12)9V.99' INIT(' '),  5 O\_CNP\_OTHER\_IS\_DFLT CHAR(001) INIT(' '),  5 O\_CNP\_OTHER\_LMT PIC'(12)9V.99' INIT(' '),  5 O\_CNP\_TOTAL\_IS\_DFLT CHAR(001) INIT(' '),  5 O\_CNP\_TOTAL\_LMT PIC'(12)9V.99' INIT(' '),  5 O\_CLS\_CL\_TRF\_CODE CHAR(010) INIT(' '),  5 O\_CLS\_CL\_TRF\_VALUE CHAR(024) INIT(' '),  5 O\_CLS\_CRD\_TRF\_CODE CHAR(011) INIT(' '),  5 O\_CLS\_CRD\_TRF\_VALUE CHAR(024) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | Ν/Α |
| **CRITICAL BOOLEAN INDICATORS** | MORE1 : CONTROLS THE LOOP OF THE INPUT RECORDS  DATA\_PHYSICAL\_OK : TURNED ON BY SERVICE CALL SRVCKSF  DATA\_LEGAL\_OK : WHEN CLIENT HAS FULL DATA |
| **SERVICES** | @SCDGLP: RETRIEVE THE OPERATION ID  @SRVCKSF:RETRIEVES PHYSICAL DATA  @SELORCN: CHECKS IF THE ACCOUNT CAN BE CONNECTED TO A DEBIT CARD  @SRVINMF: RETRIEVES COMPANY LEGAL ENTITY DATA  @SRVIXAE: RETRIEVE ALL ACCOUNT RELATIONSHIPS WITH CLIENTS  @SRVUMAS:CHECKS IF THE ACCOUNT EXISTS AND RETRIEVES PRODUCT CODE  @SRVIIPS: RETRIEVES FULL DATA LEGAL  @SCDISP: RETRIEVES MASS ISSUANCE PARAMETERIZATION  @SRVPROD: CHECKS IF THE ACCOUNT IS SAVINGS OR CURRENT |
| **Main Process Logic** | THE INPUT FILE CONTAINS CARDHOLDER DETAILS IN ORDER TO CREATE DEBIT CARD THROUGH MASSIVE ISSUANCE PROCESSING. THE OUTPUT FILE CONTAINS WITH PARAMETERIZATION DATA OF THE CARDHOLDER THAT WILL BE SENT TO WAY4 FOR THE DEBIT CARD CREATION. |
| **PROGRAM MAIN LOOP** | THE MAIN PROCESSES OF THE PROGRAM ARE:   * OPENS ALL INPUT AND OUTPUT FILES AND READS IN THE FIRST RECORD * IN A LOOP PERFORMS THE FOLLOWING:   + INITIALIZES LOCAL VARIABLES   + RETRIEVES MASS ISSUANCE PARAMETERIZATION (SERVICE SCDISP) BASED ON THE VALUE OF VARIABLE I\_PARAM\_ID PASSED IN FROM THE INPUT RECORD READ   + RETRIEVE EMBOSSING DATA AND IF THE PRODUCT GROUP = ‘02’ RETRIEVE COMPANY FULL DATA (LEGAL ENTITY) BY INVOKING SERVICE SRVIIPS AND THEN SERVICE SRVINMF. IF FULL LEGAL DATA EXIST, RAISE INDICATOR (DATA\_LEGAL\_OK) TO TRUE   + INVOKE SERVICE SRVIIPS OF SIDIPEL TO RETRIEVE PRIVATE CUSTOMER DATA (FULL CUSTOMER PHYSICAL DATA) AND IF NOT FULL DATA INVOKE SERVICE SRVINMF AND IF FULL DATA EXIST RAISE INDICATOR DATA\_PHYSICAL\_OK TO TRUE   + INVOKE SERVICE SRVUMAS TO CHECK IF THE ACCOUNT EXISTS AND RETRIEVE PRODUCT CODE   + IF BOTH PHYSICAL AND LEGAL DATA EXITS (DATA\_PHYSICAL\_OK & DATA\_LEGAL\_OK)     - INVOKE SERVICE SCDGLP TO RETRIEVE THE OPERATION ID     - CHECK IF ACCOUNT EXISTS AND RETRIEVE THE PRODUCT CODE BY INVOKING SERVICE SRVUMAS     - INVOKE SERVICE SRVPROD TO CHECK IF THE ACCOUNT IS A SAVINGS OR CURRENT     - INVOKE SERVICE SELORCN TO CHECK IF THE ACCOUNT CAN BE CONNECTED TO A DEBIT CARD     - CHECK ALSO IF CLIENT IS THE FIRST DECLARED IN THE ACCOUNT     - IN ALL CASES THAT THERE IS AN ERROR/MISSING DATA CONDITION DATA CONCERNING THE SPECIFIC PROCEDURE ARE WRITTEN TO OUTPUT FILE OUTFILE1   + READ NEXT RECORD   CLOSE FILES AND PRINT STATISTICS |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A (EXCEPT THOSE SPECIFIED INSIDE THE SERVICES) |

## [Stip Balance]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Stip Balance |
| **Batch Flow Title** | KAMPDREF |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | Way4 |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   Stip Balance process is used for the update of account balances of the connected debit cards in the card management system Way4, for all stand in transactions which don’t have communication with CBS. |

### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KAMPDREF] Stip Balance - Refresh file with Balances for WAY4

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| KAMPDREF | KALLOC | KATP.KA7GPOUT.SEQ  KATP.KA7GSOUT.SEQ  KATP.KA7GEXOF.SEQ  CRDP.PD.PBFFL.SEMT  KATP.FULLRFR2.SEQ |  | CATALOG (DISP=CATLG) FILES USING IBM UTILITY IEFBR14 |
|  | SORTA1 | CRDP.PD.PBFFL.FTIP.KAMREF01.SEQ | CRDP.PD.PBFFL.SEMT.KAMREF01.SEQ | SORT INPUT BASED ON CARD (1-16) |
|  | KA7GP | CRDP.PD.PBFFL.SEMT.KAMREF01.SEQ | KATP.KA7GPOUT.SEQ  KATP.KA7GEXOF.SEQ | CREATES OUTPUT FILES … |
|  | KA7GS | KATP.KA7GPOUT.SEQ | KATP.KA7GSOUT.SEQ | SORTS INPUT PER BRANCH AND ACCOUNT |
|  | KA6PP | KATP.KA7GSOUT.SEQ | KATP.FULLRFR2.SEQ | CREATES FILE FOR WAY4 REFRESH |
|  | DEALLOC | KATP.KA7GPOUT.SEQ  KATP.KA7GSOUT.SEQ  CRDP.PD.PBFFL.SEMT.KAMREF01.SEQ |  | DE-ALLOCATES/DELETES FILES USING IBM UTILITY IEFBR14 |

#### 

#### [KA7GP] CREATE OUTPUT FILE WITH ACCOUNTS WITH CLOSING DATE GREATER THAN OR EQUAL TO PREVIOUS WORKING DATE OF THE CURRENT DATE

| **IL Process** | |
| --- | --- |
| **Process Name** | IL MAINFRAME |
| **INPFILE1 FILE DECLARATION** | DCL KA7GP1I FILE RECORD INPUT;  DCL 1 DEPACCS,  2 CDEPACC CHAR(16) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | DCL KA7GP1O FILE RECORD OUTPUT;  DCL 1 RPINAK,  2 HBRANCH CHAR(5) INIT(' '),  2 HACC CHAR(16) INIT(' '),  2 HPROG FIXED DECIMAL (1,0) INIT(0), /\*1=ΤΑΜ,2=ΟΨΕΩΣ\*/  2 HLEDGER FIXED DECIMAL (15,2) INIT(0), /\*ΛΟΓ/ΚΟ ΥΠΟΛ. \*/  2 HHOLDCHE FIXED DECIMAL (15,2) INIT(0), /\*ΔΕΣΜ. ΕΠ/ΓΕΣ \*/  2 HHOLDDEP FIXED DECIMAL (15,2) INIT(0), /\*ΛΟΙΠΕΣ ΔΕΣΜ. \*/  2 HOVERDR FIXED DECIMAL (15,2) INIT(0), /\*OVERDRAFT \*/  2 HDEXPOVER CHAR(10) INIT(' '), /\*ΛΗΞ OVERDRAFT\*/  2 HPOOL\_ACC\_NBR CHAR(16) INIT(' '); /\*CRED ACCOUNT\*/  DCL KA7GP2O FILE RECORD OUTPUT;  DCL 1 RKA7GP2O,  2 DDATE CHAR(8) INIT(' '),  2 TTIME CHAR(6) INIT(' ');  DCL KA7GP3O FILE RECORD OUTPUT;  DCL 1 RKA7GP3O,  2 LOG\_EX CHAR(16) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | CREATES THREE OUTPUT FILES:   * KA7GP1O : CLOSING DATE '31.12.9999' & POOL\_ACCOUNT\_SYST ^= 9 & ^= 8 * KA7GP2O: CONTAINS CURRENT DATE & TIME ONLY * KA7GP3O: CONTAINS D\_CLOSE = '31.12.9999 'AND POOL\_ACCOUNT\_SYST = 9 ! = 8 AND DEPACCS.CDEPACC = '1111111111' |
| **PROGRAM MAIN LOOP** | OPENS OUTPUT FILE KA7GP2O AND WRITES IN IT CURRENT DATA AND TIME AND CLOSES IT.  FIND A WORKING DATE THAT IS PREVIOUS OF THE CURRENT DATE  OPENS INPUT FILE (KA7GP1I), READS ONE RECORD AND IN A LOOP:   * RETRIEVES ACCOUNT FORM INPUT RECORD * SELECTS FORM TABLE KATT.PREV\_DAY\_ACC\_MASTER THE ACCOUNT THAT APPEARS ON A DATE GREATER THAN OR EQUAL TO THE PREVIOUS WORKING DATE OF THE CURRENT * IF D\_CLOSE = '31.12.9999' AND POOL\_ACCOUNT\_SYST ^= 9 & ^= 8 THEN INITIALIZE HPOOL\_ACC\_NBR AND WRITE IT TO KA7GP1O ELSE * IF D\_CLOSE = '31.12.9999 'AND POOL\_ACCOUNT\_SYST = 9 ! = 8 AND DEPACCS.CDEPACC = '1111111111' WRITE TO FILE KA7GP3O   AT END OF THE LOOP (EOF OF INPUT) CLOSE FILES |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

#### 

#### [KA6PP] CREATE A FULL REFRESH FILE FOR BASE 24

| **IL Process** | |
| --- | --- |
| **Process Name** |  |
| **INPFILE1 FILE DECLARATION** | DCL KA6PP1I FILE RECORD INPUT;  DCL 1 RKA6PP1I,  2 HBRANCH CHAR(5) INIT(' '),  2 HACC CHAR(16) INIT(' '),  2 HPROG FIXED DECIMAL (1,0) INIT(0),  2 HLEDGER FIXED DECIMAL (15,0) INIT(0),  2 HHOLDCHE FIXED DECIMAL (15,0) INIT(0),  2 HHOLDDEP FIXED DECIMAL (15,0) INIT(0),  2 HOVERDR FIXED DECIMAL (15,0) INIT(0),  2 HDEXPOVER CHAR(10) INIT(' '),  2 HPOOL\_ACC\_NBR CHAR(16) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | DCL 1 RKA6PP2O\_FH,  2 FH\_REC\_FORMAT CHAR(3) INIT('FDH'),  2 FH\_REC\_SEQ PIC'(8)9' INIT(1),  2 FH\_REC\_TYP CHAR(1) INIT('F'),  2 FH\_DATETIME CHAR(14) INIT('00010101010101'),  2 FH\_CURR\_CDE CHAR(3) INIT('978'),  2 FH\_DOMAIN CHAR(32) INIT(' '),  2 FH\_FILLER CHAR(287) ;  DCL 1 RKA6PP2O\_FD,  2 FD\_REC\_FORMAT CHAR(3) INIT('CDR'),  2 FD\_REC\_SEQ PIC'(8)9' INIT(1),  2 FD\_ACCOUNT CHAR(32) INIT(' '),  2 FD\_filler CHAR(04) INIT('0000'),  2 FD\_AVAIL\_BAL PIC'(14)9' INIT(0),  2 FD\_LAST\_AUTH\_ID CHAR(32) INIT(' '),  2 FD\_ADD\_DATA CHAR(255) INIT(' ');  DCL 1 RKA6PP2O\_FT,  2 FT\_REC\_FORMAT CHAR(3) INIT('FTR'),  2 FT\_REC\_SEQ PIC'(8)9' INIT(1),  2 FT\_FILLER CHAR(4) INIT('0000'),  2 FT\_CHECKSUM PIC'(14)9' INIT(0),  2 FT\_FILLER1 CHAR(319) ;  DCL 1 RKA6PP2O,  2 RECFIELD CHAR(348) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | CRDPOVR: NEEDS TO BE INVESTIGATED-SOURCE FILE |
| **Main Process Logic** | READS INPUT FILE (KA6PP1I) AND FOR EVERY ACCOUNT DETERMINES THE AVAILABLE BALANCE AND WRITES IT TO THE OUTPUT FOR WAY4/BASE24 REFRESH |
| **PROGRAM MAIN LOOP** | OPENS ONE INPUT (KA6PP1I) AND TWO OUTPUT FILES(KA6PP1O/KA6PP2O)  WRITES DATE/TIME AS HEADERS IN KA6PP2O  IN A LOOP TILL INPUT FILE REACHES EOF CONDITION   * CONCATENATES BRANCH PLUS ACCOUNT PLUS CHECK DIGIT OF THE ACCOUNT * HANDLES (IN AN APPROPRIATE WAY OVERFLOW CONDITIONS)!!! * DETERMINES THE AVAILABLE BALANCE * DETERMINES THE OVERDRAFT LIMIT (EXCESSIVE TO ACCOUNT AVAILABLE AMOUNT ALLOWED BY BANK) * WRITES DATA TO OUTPUT FILE (KA6PP1O) * FILLS IN DATA AREA FOR STIP BALANCE DETAIL RECORD AND WRITES DATA TO THE OUTPUT FILE (KA6PP2O) * INITIALIZES INTERMEDIATE VARIABLE * READS NEXT RECORD   CLOSES INPUT & OUTPUT FILES |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

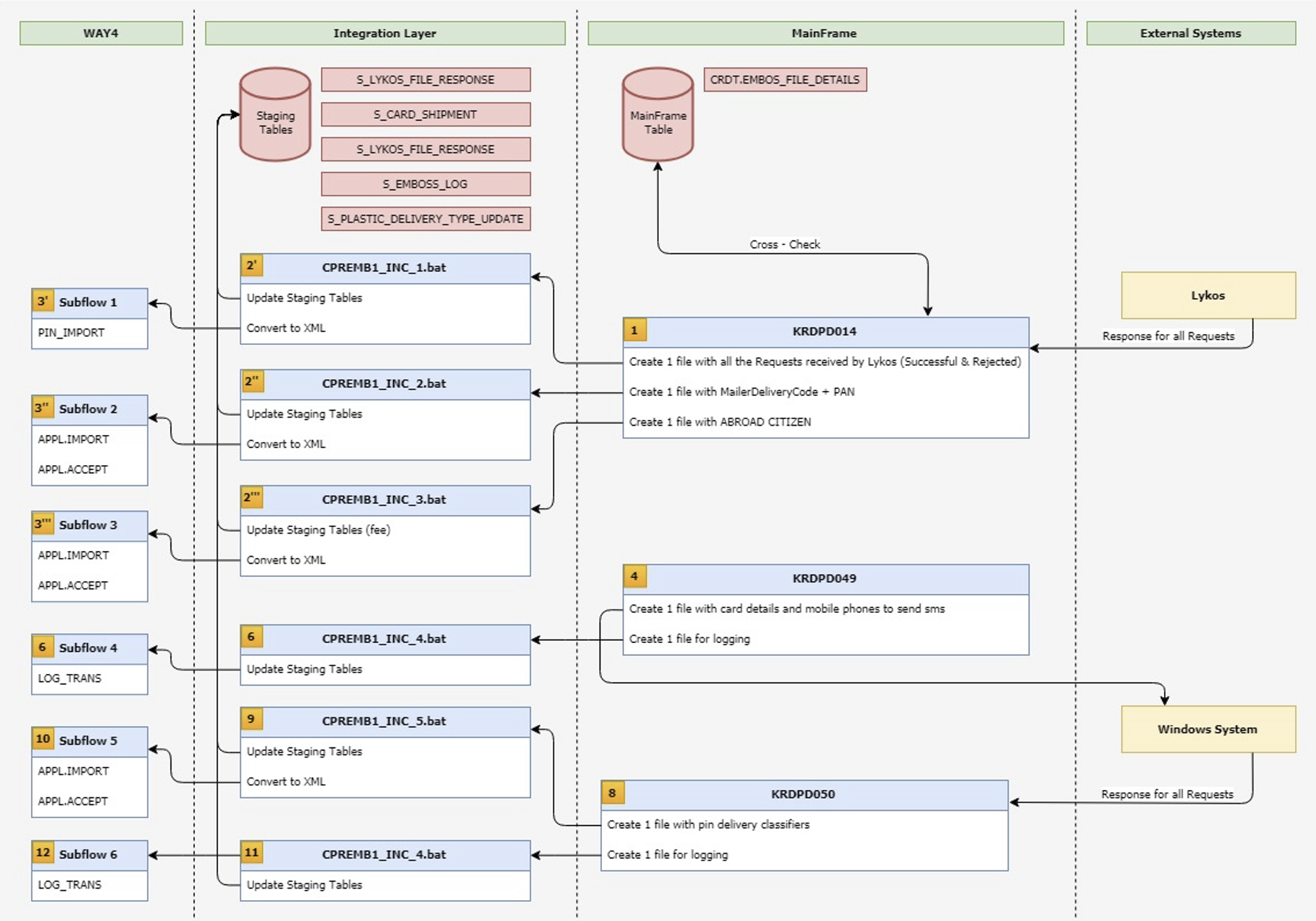
## 

## [Embossing Incoming]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Embossing Incoming |
| **Batch Flow Title** | KRDPD014 |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | Way4 |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   The embossing incoming process is being used to process the response file retrieved from Inform Lykos about all the cards that have been produced by them. It uses clients table for validations as well as cards tables. |



### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KRDPD014] Embossing Incoming

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| KRDPD014 | ALLOCAT | N/A | CRDP.PD.LKSRS.SEMT.CRD01406.SEQ  CRDP.PD.LKSRS.SEMT.CRD01407.SEQ  CRDP.PD.LPREP.FTOP.CRD06500.SEQ  CRDP.PD.LPREP.FTOP.CRD06501.SEQ | ALLOCATE FILES USING IBM UTILITY IEFBR14 OPTION DISP=(NEW,CATLG) |
|  | K18CS1 | KATP.PD.PREPD.FTIP.EMBORESP.SEQ | CRDP.PD.LKSRS.SEMT.CRD01406.SEQ | SORT ONLY DUAL REQUESTS |
|  | K18CS2 | KATP.PD.PREPD.FTIP.EMBORESP.SEQ | CRDP.PD.LKSRS.SEMT.CRD01407.SEQ | SORT REQUESTS EXCEPT DUAL |
|  | K18CS3 | CRDP.PD.LKSRS.SEMT.CRD01407.SEQ  CRDP.PD.LKSRS.SEMT.CRD01406.SEQ | KATP.PD.PREPD.FTIP.EMBORESP.SEQ | MERGE TWO FILES (DUAL&ALL) |
|  | K18CS4 | KATP.PD.PREPD.FTIP.EMBORESP.SEQ | CRDP.PD.LKSRS.FTIP.CRD01401.SEQ | COPY-REFORMAT FILES |
|  | K18CP | CRDP.PD.LKSRS.FTIP.CRD01401.SEQ | CRDP.PD.LKSRS.SEMT.CRD01402.SEQ | INSERT NEW RECORD TO KCCT.TRACK\_INFO TABLE AND EXPORTS A NEW FILE INCLUDING ALL NECESSARY DATA FOR PIN MANAGEMENT |
|  | K18CS2 | CRDP.PD.LKSRS.SEMT.CRD01402.SEQ | CRDP.PD.LKSRS.FTOP.CRD01403.SEQ | SUPPLEMENT FILE WITH HEADER AND TRAILER RECORDS |
|  | K21CP | CRDP.PD.LKSRS.FTIP.CRD01401.SEQ | CRDP.PD.LKSRS.SEMT.CRD01404.SEQ (K21CP1)  CRDP.PD.LKSRS.SEMT.CRD01405.SEQ (K21CP2) | **K21CP1** WILL CONTAIN ALL CARDS THAT HAVE BEEN RECEIVED BY  LYKOS AND WAY4 SHOULD UPDATE THEIR PLASTIC DELIVERY TYPE.  **K21CP2** WILL CONTAIN ALL CARDS THAT HAVE BEEN RECEIVED BY  LYKOS AND WHOSE CARDHOLDER IS AN ABROAD RESIDENT. |
|  | K21CS1 | CRDP.PD.LKSRS.SEMT.CRD01404.SEQ | CRDP.PD.LKSRS.FTOP.CRD01406.SEQ | SUPPLEMENT FILE WITH HEADER AND TRAILER RECORDS |
|  | K21CS2 | CRDP.PD.LKSRS.SEMT.CRD01405.SEQ | CRDP.PD.LKSRS.FTOP.CRD01407.SEQ | SUPPLEMENT FILE WITH HEADER AND TRAILER RECORDS |
|  | K61DP |  | CRDP.PD.LPREP.FTOP.CRD06500.SEQ  CRDP.PD.LPREP.FTOP.CRD06501.SEQ | TO BE COMPLETED LATER |
|  | K67CP | CRDP.PD.LKSRS.FTOP.CRD01407.SEQ | CRDP.PD.FCARD.FTOP.CRD01409.SEQ | SUPPLEMENT FILE WHOSE CARDHOLDER IS AN ABROAD RESIDENT WITH EXTRA FIELDS |
|  | DEALLOC |  | CRDP.PD.LKSRS.SEMT.CRD01402.SEQ  CRDP.PD.LKSRS.SEMT.CRD01404.SEQ  CRDP.PD.LKSRS.SEMT.CRD01405.SEQ  CRDP.PD.LKSRS.SEMT.CRD01406.SEQ  CRDP.PD.LKSRS.SEMT.CRD01407.SEQ | DEALLOCATE FILES USING IBM UTILITY IEFBR14 (**DISP=(OLD,DELETE**)) |
|  | CLEARFTP |  | CRDP.PD.LKSRS.FTIP.CRD01401.SEQ | CLEAR OUT FILE |

#### [K18CP] REFORMATS LYKOS RESPONSE FILE TO BE FED TO WAY4

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | **K18CPI**  DCL 1 RK18CPI,  2 REC\_NO CHAR (06) INIT(' '),  2 FILLER1 CHAR (01) INIT(' '),  2 CARD\_EMB CHAR (26) INIT(' '),  2 FILLER2 CHAR (01) INIT(' '),  2 CARD\_NO CHAR (16) INIT(' '),  2 FILLER3 CHAR (01) INIT(' '),  /\* 20230530 START\_CODE U714 DUAL/COMBO CARD \*/  2 COMBO\_CARD CHAR (16) INIT(' '),  /\* 20230530 END\_CODE U714 DUAL/COMBO CARD \*/  2 FILLER4 CHAR (01) INIT(' '),  2 REJ\_CODE CHAR (03) INIT(' '),  2 FILLER5 CHAR (01) INIT(' '),  2 REJ\_DESCR CHAR (100) INIT(' '),  2 FILLER6 CHAR (01) INIT(' '),  2 FILE\_NAME CHAR (08) INIT(' '),  2 FILLER7 CHAR (01) INIT(' '),  2 ELTA\_DATE CHAR (10) INIT(' '),  2 FILLER8 CHAR (01) INIT(' '),  2 ACT\_PLS\_DEL\_TYPE CHAR (02) INIT(' '),  2 FILLER9 CHAR (01) INIT(' '),  2 FNL\_PLS\_DEL\_TYPE CHAR (02) INIT(' '),  2 FILLER10 CHAR (01) INIT(' '),  /\*20220913 START\_CODE E.PAPPAS U714 - CC TRACKING\*/  2 TRACKING\_NUMBER CHAR (20) INIT(' '),  /\*20220913 END\_CODE E.PAPPAS U714 -CC TRACKING\*/  2 FILLER11 CHAR (01) INIT(' '),  2 EXCEPT\_FLG CHAR (01) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | **K18CPO**  DCL 1 RK18CPO,  2 O\_REC\_NO CHAR (06) INIT(' '),  2 O\_CARD\_EMB CHAR (26) INIT(' '),  2 O\_CARD\_NO CHAR (16) INIT(' '),  2 O\_FILLER CHAR (02) INIT(' '),  2 O\_REJ\_CODE CHAR (03) INIT(' '),  2 O\_REJ\_DESCR CHAR (100) INIT(' '),  2 O\_FILE CHAR (15) INIT(' '),  2 O\_ELTA\_DATE CHAR (10) INIT(' '),  2 O\_ORIG\_REC\_ID CHAR (06) INIT(' '),  2 O\_EXCEPT\_FLG CHAR (01) INIT(' '),  2 O\_PLS\_DEL\_TYPE CHAR (02) INIT(' '),  2 WAY4\_FILE\_NAME CHAR (40) INIT(' '),  2 TRANSACTION\_ID CHAR (255) INIT(' '),  2 BATCH\_ID CHAR (32) INIT(' '),  /\*20220913 START\_CODE E.PAPPAS U714 - CC TRACKING\*/  2 O\_COMBO\_CARD CHAR (16) INIT(' ');  /\*20220913 END\_CODE E.PAPPAS U714 -CC TRACKING\*/ |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | SRVIXAS: CHECK IF CARD HAS SIDIPEL  KCCSPN: DELETE ENTRY IF CARD HAS SMARTPIN  UCCSPN: UPDATE KCCT.SMARTPIN WITH NEW FLAG  UCDEFD2: UPDATE CRDT.EMBOS\_FILE\_DETAIL TABLE  SCDTRC: RETRIEVE TRACK INFO FROM TABLE KCCT.TRACK\_INFO  ICDTRC: INSERT ROW TRACK INFO  SCDEFD2: RETRIEVE INFORMATION FROM CRDT.EMBOS\_FILE\_DETAIL  SCDEFD1: CHECK IF CARD IS DEBIT AND RETURN INDICATOR TRUE/FALSE  SCCSPN: CHECKS IF CARD HAS SMARTPIN  SCCSPN1: CHECKS IF FLG\_DELIVERY TYPE IS VIA SMS |
| **Main Process Logic** | PROCESS INPUT FILE RECEIVED FROM LYKOS BY APPENDING DATA CONCERNING PIN MANAGEMENT FOR EVERY CARD |
| **PROGRAM MAIN LOOP** | THE MAIN STEPS THAT THE PROGRAM PERFORMS ARE:   * GETS CURRENT DATE/TIME, OPENS INPUT & OUTPUT FILES AND READS 1ST RECORD * IN A LOOP TILL THE EOF OF INPUT FILE, THE FOLLOWING PROCESSES ARE CARRIED OUT   + IGNORE ALL NON DEBIT CARDS AND DEBIT CYPRUS CARDS(589893)   + IF CARD BIN = 527801 RETRIEVE TRACK INFO FROM TABLE KCCT.TRACK\_INFO BY INVOKING SERVICE SCDTRC   + CHECK IF CARD HAS SIDIPEL BY INVOKING SERVICE SRVIXAS AND IF SO,     - CHECK IF CARD HAS SMARTPIN BY INVOKING SERVICE SCCSPN AND IF SO DELETE IT BY INVOKING SERVICE KCCSPN     - IF NOT EQUALS KCCT.TRACK\_INFO AND IN LYKOS RESPONSE FILE (^L\_DOUBLE\_FLG) THEN CHECK IF THE TRACKING NUMBERS ARE DIFFERENT (L\_DIF\_TRCNUM\_FLG) AND IF SO UPDATE TABLE TRACK\_INFO   + IF ALL THE ABOVE CONDITIONS ARE MET, POPULATE STRUCTURE RK18CPO AND WRITE OUTPUT FILE (K18CPO)   + PROCEED TO READING THE NEXT RECORD * AT END OF INPUT FILE CLOSE ALL FILES AND PRINT STATISTICS |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | ONLY THE TABLES REFERRED IN SERVICES |

#### 

#### [K21CP] BASED ON FILE RECEIVED FROM LYKOS, CREATE TWO FILES THAT WILL UPDATE WAY4 WITH MAILER CODE AND DEBIT CARDS OF ABROAD RESIDENTS

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | RK21CPI (LYKOS RESPONSE)  DCL 1 RK21CPI,  2 REC\_NO CHAR (006) INIT(' '),  2 FILLER1 CHAR (001) INIT(' '),  2 CARD\_EMB CHAR (026) INIT(' '),  2 FILLER2 CHAR (001) INIT(' '),  2 CARD\_NO CHAR (016) INIT(' '),  2 FILLER3 CHAR (001) INIT(' '),  /\* 20230530 START\_CODE U714 DUAL/COMBO CARD \*/  2 COMBO\_CARD CHAR (016) INIT(' '),  /\* 20230530 END\_CODE U714 DUAL/COMBO CARD \*/  2 FILLER4 CHAR (001) INIT(' '),  2 REJ\_CODE CHAR (003) INIT(' '),  2 FILLER5 CHAR (001) INIT(' '),  2 REJ\_DESCR CHAR (100) INIT(' '),  2 FILLER6 CHAR (001) INIT(' '),  2 FILE CHAR (008) INIT(' '),  2 FILLER7 CHAR (001) INIT(' '),  2 ELTA\_DATE CHAR (010) INIT(' '),  2 FILLER8 CHAR (001) INIT(' '),  2 ACT\_PLS\_DEL\_TYPE CHAR (002) INIT(' '),  2 FILLER9 CHAR (001) INIT(' '),  2 FNL\_PLS\_DEL\_TYPE CHAR (002) INIT(' '),  2 FILLER10 CHAR (001) INIT(' '),  /\*20220913 START\_CODE E.PAPPAS U714 - CC TRACKING\*/  2 TRACKING\_NUMBER CHAR (020) INIT(' '),  /\*20220913 END\_CODE E.PAPPAS U714 -CC TRACKING\*/  2 FILLER11 CHAR (001) INIT(' '),  2 EXCEPT\_FLG CHAR (001) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | K21CP1: (PLUS INCLUDES PLASTIC DELIVERY TYPE AND UNIQUE MAILER CODE)  DCL 1 RK21CP1,  2 O\_CARD\_NO CHAR (016) INIT(' '),  2 O\_FILE CHAR (015) INIT(' '),  2 O\_ORIG\_REC\_ID CHAR (006) INIT(' ');  K21CP2: DEBITS OF ABROAD RESIDENTS–EXTRA FEE  DCL 1 RK21CP2,  2 O\_CARD\_NO CHAR (016) INIT(' '),  /\*20230609 START\_CODE E.PAPPAS U714 - DUAL/COMBO\*/  2 O\_CARD\_NO2 CHAR (016) INIT(' ');  /\*20230609 END\_CODE E.PAPPAS U714 - DUAL/COMBO\*/ |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | READ\_K21CPI:CONTROLS EOF OF INPUT FILE  L\_IS\_DEBIT: RAISED WHEN DEBIT POPULATED WITH RELATED DEBIT CARD DATA  L\_CARD\_FOUND:RAISED WHEN CARDHOLDER IS A RESIDENT ABROAD  L\_CRD\_FND: RAISED TO INDICATE EMBOSSING DATA EXIST FOR THE CARD |
| **SERVICES** | SCDEFD1: INVOKED TO INDICATE IF A CARD IS A DEBIT ONE AND POPULATE WITH DATA  SCDEFD2:INVOKED TO RETRIEVE EMBOSSING DATA FROM CRDT.EMBOS\_FILE\_DETAIL |
| **Main Process Logic** | BASED ON INPUT FILE FROM LYKOS TWO OUTPUT FILES ARE CREATED THAT WILL BE FED TO WAY4, ONE WITH INFO AS TO DELIVERY TYPE AND UNIQUE MAILER CODE AND ANOTHER WITH EMBOSSING DATA FOR CARDHOLDERS WITH ABROAD RESIDENCE |
| **PROGRAM MAIN LOOP** | BASED ON INPUT FILE FROM LYKOS TWO OUTPUT FILES ARE CREATED THAT WILL BE FED TO WAY4  MORE SPECIFICALLY THE PROGRAM DOES (IN SEQUENCE):   * GET THE CURRENT DATE/TIME AND OPEN ALL FILES, READ 1ST INPUT RECORD * PERFORM IN A LOOP THE FOLLOWING STEPS:   + IF CARD’S BIN IS NOT 589893 CHECK IF IT IS A DEBIT CARD BY INVOKING SERVICE SCDEFD1 AND IF SO WRITE TO FILE K21CP1 ALONG WITH INFO FOR PLASTIC DELIVERY TYPE AND UNIQUE MAILER CODE FOR UPDATE OF WAY4. IN ADDITION TO THE ABOVE CONDITIONS INVOKE SERVICE SCDEFD2 TO RETRIEVE CARD INFO FROM CRDT.EMBOS\_FILE\_DETAIL AND IF SO POPULATE OUTPUT AREA AND WRITE FILE K21CP2   + REPEAT LOOP BY READING NEXT INPUT CARD AND AT END * AT END CLOSE ALL FILES AND PRINT STATISTICS |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | NO TABLES EXCEPT OF THOSE INSIDE THE SERVICES |

## 

#### [K61DP] CREATES TWO OUTPUT FILES WITH TRACK DATA BASED ON A SPECIFIC QUERY

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | TABLES: KCCT.TRACK\_INFO |
| **OUTFILE1 FILE DECLARATION** | **K61DPOUT**  DCL 1 RK61DP,  2 BARCODE CHAR(20) INIT(' '), /\* TRACKING NUMBER \*/  2 FILLER1 CHAR(01) INIT(';'),  2 C\_REF CHAR(30) INIT(' '), /\* SYDIPEL \*/  2 FILLER2 CHAR(01) INIT(';'),  2 R\_NAME CHAR(150) INIT(' '),  2 FILLER3 CHAR(01) INIT(';'),  2 R\_ADDRESS CHAR(150) INIT(' '),  2 FILLER4 CHAR(01) INIT(';'),  2 CITY CHAR(40) INIT(' '),  2 FILLER5 CHAR(01) INIT(';'),  2 ZIPCODE CHAR(05) INIT(' '),  2 FILLER6 CHAR(01) INIT(';'),  2 PHONE\_STATH CHAR(10) INIT(' '),  2 FILLER7 CHAR(01) INIT(';'),  2 PHONE\_MOBILE CHAR(10) INIT(' '),  2 FILLER8 CHAR(01) INIT(';'),  2 YPHRESIA CHAR(03) INIT('200'),  2 FILLER9 CHAR(01) INIT(';'),  2 WEIGHT CHAR(10) INIT('0000000100'),  2 FILLER10 CHAR(01) INIT(';'),  2 TROPOS\_EPIDOSIS CHAR(01) INIT('2'); /\* ΜΔ \*/  **K61DPOU2**  DCL 1 RK61DP2, /\* LRECL = 6630 \*/  2 APPLICATION CHAR(30) INIT(' '),  2 FIELD(12),  5 FIELDNAME CHAR(50) INIT(' '),  5 FIELDVALUE CHAR(500) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | DCL 1 RK61DP2, /\* LRECL = 6630 \*/  2 APPLICATION CHAR(30) INIT(' '),  2 FIELD(12),  5 FIELDNAME CHAR(50) INIT(' '),  5 FIELDVALUE CHAR(500) INIT(' '); |
| **CRITICAL BOOLEAN INDICATORS** | ROW\_ERR: USED FOR TRACKING AN ERROR ANYWHERE  WS\_PHONE\_FND:INDICATES THE EXISTENCE OF PHONES |
| **SERVICES** | SRVIXAS:FINDS SIDIPEL FROM A CARD  SRVIIPS:FINDS THE NAME FROM SIDIPEL  SRVCLAF:FINDS PHONES |
| **Main Process Logic** | BASED ON THE STRUCTURE OF A SPECIFIC QUERY, RETRIEVES AND EXPORTS CARD TRACK DATA |
| **PROGRAM MAIN LOOP** | THE MAIN TASKS THAT ARE PERFORMED ARE:   * OPEN OUTPUT FILES K61DPOUT & K61DPOU2 * OPEN CURSOR WITH TRACK DATA FROM KCCT.TRACK\_INFO AND PROD\_STATUS IN ('5','4') (RESPONSE FROM LYKOS) * INITIALIZE VARIABLES * IN A LOOP (**RECOMMENDATION**:THE FIRST FETCH SHOULD BE PLACED OUTSIDE THE LOOP LIKE READING A NORMAL FILE–CURSOR IS NOT DIFFERENT THAN A NORMAL FILE AND INSIDE THE LOOP ANOTHER FETCH AS A LAST STATEMENT OF THE INLINE DO)   + FETCH DATA INTO DCLTRACK\_INFO   + INVOKE SERVICE SRVIXAS TO GET SIDIPEL FROM CARD   + INVOKE SERVICE SRVIIPS AND GET NAME FROM SIDIPEL   + INVOKE SERVICE SRVCLAF AND GET PHONE NUMBERS   + DETERMINE IF IT IS ABOUT:RENEWAL/REPLACEMENT PLASTIC/NEW CARD   + IF THERE WAS NO ERRORS FROM THE ABOVE OUTPUT DATA TO FILE K61DPOUT, POPULATE AND WRITE DATA TO K61DPOU2   + FETCH NEXT CURSOR ROW   AT END, (SQLCODE=100) CLOSE CURSOR AND OUTPUT FILES |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | KCCT.TRACK\_INFO (TRACKING DATA) |

## 

#### [K67CP] SUPPLEMENT INPUT FILE WITH ADDITIONAL FIELDS OF AN ABROAD CUSTOMER

| **IL Process** | |
| --- | --- |
| **Process Name** |  |
| **INPFILE1 FILE DECLARATION** | DCL 1 I\_REC,  5 REC\_TYPE CHAR (02) INIT(' '),  5 CARD CHAR (16) INIT(' '),  /\* 20230613 START\_CODE U714 DUAL/COMBO CARD \*/  5 COMBO\_CARD CHAR (16) INIT(' '),  /\* 20230613 END\_CODE U714 DUAL/COMBO CARD \*/  5 FILLER CHAR (08) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | DCL 1 O\_REC,  5 CARD CHAR (19) INIT(' '),  5 EMBOSSING\_NAME CHAR (26) INIT(' '),  5 SNAME CHAR (40) INIT(' '),  5 FNAME CHAR (40) INIT(' '),  5 PNAME CHAR (40) INIT(' '),  5 STREET CHAR (40) INIT(' '),  5 CITY CHAR (40) INIT(' '),  5 ZIPCODE CHAR (40) INIT(' '),  5 COUNTRY CHAR (40) INIT(' '),  5 LYKOS\_FILE\_NAME CHAR (15) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_INPUT: CONTROL LOOP OF INPUT FILE |
| **SERVICES** | N/A |
| **Main Process Logic** | SUPPLEMENT INPUT FILE WITH ADDITIONAL FIELDS OF AN ABROAD CUSTOMER FOR EMBOSSING. TABLES USED CRDT.EMBOS\_FILE\_DETAIL & CRDT.EMBOS\_FILE\_MASTER |
| **PROGRAM MAIN LOOP** | THE MAIN STEPS IN THIS PROGRAM ARE:   * OPENS INPUT & OUTPUT FILES AND READS THE 1ST RECORD * IN A LOOP UNTIL EOF PERFORMS IN SEQUENCE (PROCESSES ONLY DETAIL RECORDS ‘FD’   + RETRIEVE FROM TABLES CRDT.EMBOS\_FILE\_DETAIL JOINED WITH CRDT.EMBOS\_FILE\_MASTER CARD DATA   + WRITE OUTPUT FILE (K67CPO)   + READ NEXT RECORD AND REPEAT THE SAME PROCESS TILL EOF   AT END, CLOSE FILES AND PRINT STATISTICS |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER |

## 

## [Deletion of card]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Deletion of card |
| **Batch Flow Title** | KRDPD022 |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | Way4 |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   The deletion of card process retrieves a file with cards that are permanently closed in W4 in order to delete the client card relation. It also uses a card service that detects if a card its migrated or not. |

### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KRDPD022] Deletion of card

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| KRDPD022 | K25CP | CRDP.PD.DELCC.FTIP.CRD02201.SEQ |  | Validate file consistency of file that includes cards permanently closed/automatically in Way4 |
|  | K22CP | CRDP.PD.DELCC.FTIP.CRD02201.SEQ |  | DELETION OF CARD/CUST RELATION |

#### [K25CP] Check the input file for the next action “Deletion of card from SYDIPEL”

| **IL Process** | |
| --- | --- |
| **Process Name** |  |
| **INPFILE1 FILE DECLARATION** | DCL 1 RECFIELD,  5 ROWCODE CHAR (02) INIT(' '),  5 REST\_DETAILS CHAR (16) INIT(' ');  DCL 1 FDTL\_INPFILE1 BASED(ADDR(RECFIELD)),  5 FDTL\_ROWCODE CHAR (02) INIT(' '),  5 FDTL\_REST CHAR(16) INIT(' ');  DCL 1 FTRL\_INPFILE1 BASED(ADDR(RECFIELD)),  5 FTRL\_ROWCODE CHAR (02) INIT(' '),  5 FTRL\_TOTAL\_RECS PIC'(07)9' INIT(0),  5 FILLER CHAR (09) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | * Open input file * Read first row * Execute the following loop in next tab * Close File * Print statistics about header , trailer existence (value 1) if missing abend * Print statistics about Data ROWCODE='FD' |
| **PROGRAM MAIN LOOP** | * Check for header when found FOUND\_HEADER = 1 * Check for Data ROWCODE='FD' , increment counter FD\_COUNTER\_TOTAL * Check for Footer , when found FOUND\_TRAILER = 1 , check consistency FD\_COUNTER\_TOTAL read from those read from the trailer line FTRL\_INPFILE1.FTRL\_TOTAL\_RECS : if not equal abend program |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

#### 

#### [K22CP] Deletion of card from SYDIPEL

| **IL Process** | |
| --- | --- |
| **Process Name** | ILMainframe |
| **INPFILE1 FILE DECLARATION** | DCL 1 RECFIELD,  2 ROWCODE CHAR(02) INIT(' '),  2 REST\_DETAILS CHAR(16) INIT(' ');  DCL 1 FD\_INPFILE1 BASED(ADDR(RECFIELD)),  2 FD\_ROWCODE CHAR(02) INIT('FD'),  2 FD\_CARD\_PAN CHAR(16) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | @SRV44NP : Called to Define MIGRATED & NON-MIGRATED CARDS FLAG  @SRVIXAD : Service to delete card/cust Relation |
| **Main Process Logic** | · Open input file  · Read first row  · Execute the following loop in next tab  · Close File  · Execute Statistics of program |
| **PROGRAM MAIN LOOP** | For each row that is not header or footer increment counter for Data , COUNTERS\_DEC.REC\_READ plus :   * checks if the first 6 characters of FD\_CARD\_PAN is '516297' and do nothing else   + if CARD pan in ('535142' '535143','416581','930050','535551',' 589242', '442317' , '533824' , '534604' ) do nothing else :     - Invoke service LINK\_SRV44NP to get L\_IN\_SESPIKA flag   + if CARD pan in ('535142' '535143','416581','930050','535551',' 589242', '442317' , '533824' , '534604' ) OR L\_IN\_SESPIKA = 'N' call procedure LINK\_SRVIXAD ( actually same check as above but take in consideration also the FLAG response from the SERVICE SRV44NP ) to invoke service SRVIXAD in order to delete CARD/CUSTOMER RELATION     - For the case that CARD pan in ('535142' '535143','416581','930050','535551',' 589242') assign input to the service @SRVIXAD.IDATA.I\_CLSF\_IT\_ACC\_K = 15100005 else @SRVIXAD.IDATA.I\_CLSF\_IT\_ACC\_K = 15100007     - Execute service to delete CARD/CUSTOMER RELATION * Every 100 records commits the transaction by calling COMMIT\_RTN |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

## 

## 

## [Massive Issuance/Reissuance Voucher]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Massive Loading Voucher |
| **Batch Flow Title** | KRPPD013-015 & KRPPD017-019 |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | Way4 |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   The Massive Issuance/Reissuance Voucher process uses a flat file for every organization’s voucher cards to be created. It uses a parametric card table as well as deposit services for placing a hold and the debit of the commision and client services for validations. |

### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KRPPD000] Voucher Massive Issuance (Req) use case: Validate and Enrich files

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD000** | IEFBR14 | CRDP.PD.PRVOU.SEMT.KD5RPDUP.SEQ,  CRDP.PD.PRVOU.SEMT.KD5RPUNI.SEQ,  CRDP.PD.PRVOU.SEMT.KD5RPTMP.SEQ,  CRDP.PD.PRVOU.SEMT.KD5DPOU.SEQ,  CRDP.PD.PRVOU.SEMT.KD5LPOUT.SEQ,  CRDP.PD.PRVOU.SEMT.KD6OPOUT.SEQ | N/A | File allocation |
|  | KD5DP | CRDP.PD.PRVOU.FTIP.KD5DPINP.SEQ | CRDP.PD.PRVOU.SEMT.KD5DPOU.SEQ | Validate file header |
|  | KD5LP | CRDP.PD.PRVOU.SEMT.KD5DPOU.SEQ | CRDP.PD.PRVOU.SEMT.KD5LPOUT.SEQ | Validate file detail records |
|  | KD5RP | CRDP.PD.PRVOU.SEMT.KD5LPOUT.SEQ | CRDP.PD.PRVOU.PERP.KD5RPOUT.SEQ | Enrich file with company data |
|  | ICETOOL | CRDP.PD.PRVOU.PERP.KD5RPOUT.SEQ,  CRDP.PD.PRVOU.SEMT.KD5RPTMP.SEQ | CRDP.PD.PRVOU.SEMT.KD5RPDUP.SEQ,  CRDP.PD.PRVOU.SEMT.KD5RPUNI.SEQ | Create file with records with same AFM |
|  | DEMPP | CRDP.PD.PRVOU.SEMT.KD5RPDUP.SEQ | N/A | Check whether file with duplicates isn’t empty |
|  | KD6NP | CRDP.PD.PRVOU.PERP.KD5RPOUT.SEQ | CRDP.PD.PRVOU.SEMT.KD5RPDUP.SEQ | If said files isn’t empty, add rejection message to original file |
|  | KD6OP | CRDP.PD.PRVOU.PERP.KD5RPOUT.SEQ | CRDP.PD.PRVOU.SEMT.KD6OPOUT.SEQ | Translate greek names to check suspect |
|  | KD6PP | CRDP.PD.PRVOU.SEMT.KD6OPOUT.SEQ | CRDP.PD.PRVOU.PERP.KD5RPOUT.SEQ | Check suspect |
|  | KD5SP | CRDP.PD.PRVOU.PERP.KD5RPOUT.SEQ | CRDP.PD.PRVOU.PERP.CRD00101.SEQ | Voucher: enrich file with owner data |
|  | IEFBR14 | CRDP.PD.PRVOU.SEMT.KD5RPDUP.SEQ,  CRDP.PD.PRVOU.SEMT.KD5RPUNI.SEQ,  CRDP.PD.PRVOU.SEMT.KD5RPTMP.SEQ,  CRDP.PD.PRVOU.SEMT.KD5DPOU.SEQ,  CRDP.PD.PRVOU.SEMT.KD5LPOUT.SEQ,  CRDP.PD.PRVOU.SEMT.KD6OPOUT.SEQ | N/A | File deallocation |

#### 

#### [KRPPD001] Voucher Massive Issuance (Req) use case: Merge and Validate files

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD001** | SORT | CRDP.PD.PRVOU.PERP.CRD00101.SEQ | CRDP.PD.PRVOU.SEMT.CRD00102.SEQ | Sort input file to only get header |
|  | SORT | CRDP.PD.PRVOU.PERP.CRD00101.SEQ | CRDP.PD.PRVOU.SEMT.CRD00103.SEQ | Sort input file to only get IBANS and SYDIPELs |
|  | SORT | CRDP.PD.PRVOU.SEMT.CRD00103.SEQ | CRDP.PD.PRVOU.SEMT.CRD00104.SEQ | Remove duplicate IBANs from the previous step |
|  | K79CP | CRDP.PD.PRVOU.SEMT.CRD00102.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00104.SEQ | CRDP.PD.PRVOU.SEMT.CRD00105.SEQ | Merging files |
|  | K80CP | CRDP.PD.PRVOU.SEMT.CRD00105.SEQ | CRDP.PD.PRVOU.INTT.CRD00106.SEQ | Perform validations, enrich file with INST\_DEPOSIT\_ACCOUNT, RC, RC\_TXT |

#### [KRPPD002] Voucher Massive Issuance (Req) use case: Calculate production fees

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD002** | SORT | CRDP.PD.PRVOU.PERP.CRD00101.SEQ | CRDP.PD.PRVOU.SEMT.CRD00207.SEQ | Sort file to only get details |
|  | SORT | CRDP.PD.PRVOU.INTT.CRD00106.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00207.SEQ | CRDP.PD.PRVOU.SEMT.CRD00208.SEQ | Merge files |
|  | K81CP | CRDP.PD.PRVOU.SEMT.CRD00208.SEQ | CRDP.PD.PRVOU.INTT.CRD00209.SEQ | Calculate card production fee for each valid request |

#### [KRPPD003] Voucher Massive Issuance (Req) use case: Place holds

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD003** | DSNTIAUL | N/A | CRDP.PD.PRVOU.SEMT.CRD00310.SEQ | Unload CRDT.PREPAID\_REQ\_ISS for sum amounts |
|  | K74CP | CRDP.PD.PRVOU.SEMT.CRD00310.SEQ | N/A | Place holds based on amount, fee and iban |
|  | DSNTIAUL | N/A | CRDP.PD.PRVOU.SEMT.CRD00311.SEQ | Unload CRDT.PREPAID\_REQ\_ISS for SSIS |
|  | SORT | CRDP.PD.PRVOU.SEMT.CRD00311.SEQ | CRDP.PD.PRVOU.FTOP.CRD00312.SEQ | Copy to permanent FTP file and enrich file with header and trailer |
|  | IEFBR14 | CRDP.PD.PRVOU.SEMT.CRD00102.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00103.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00104.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00105.SEQ,  CRDP.PD.PRVOU.INTT.CRD00106.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00207.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00208.SEQ,  CRDP.PD.PRVOU.INTT.CRD00209.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00310.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00311.SEQ | N/A | Deallocation of working files. |

#### [KRPPD004] Voucher Massive Issuance (Res) use case: Connect SYDIPEL & insert to SMARTPIN

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD004** | SORT | CRDP.PD.PRVOU.FTIP.CRD00400.SEQ | CRDP.PD.PRVOU.SEMT.CRD00401.SEQ | Remove header and trailer and REC\_TYPE from input file |
|  | SORT | CRDP.PD.PRVOU.SEMT.CRD00401.SEQ | CRDP.PD.PRVOU.SEMT.CRD00402.SEQ | Remove header and trailer and REC\_TYPE from input file |
|  | K82CP | CRDP.PD.PRVOU.SEMT.CRD00402.SEQ | N/A | Connect to SYDIPEL and insert into SMARTPIN tables |

#### [KRPPD005] Voucher Massive Issuance (Res) use case: Deblock and Debit/Credit Account

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD005** | DSNTIAUL | N/A | CRDP.PD.PRVOU.SEMT.CRD00503.SEQ | Unload from CRDT.PREPAID\_REQ\_ISS for sum amounts |
|  | K77CP | CRDP.PD.PRVOU.SEMT.CRD00503.SEQ | N/A | Deblock and debit/credit account |

#### [KRPPD006] Voucher Massive Issuance (Res) use case: Validate files and Archive Tables

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD006** | DSNTIAUL | N/A | CRDP.PD.PRVOU.SEMT.CRD00604.SEQ | Unload from CRDT.PREPAID\_REQ\_ISS for log |
|  | K78CP | CRDP.PD.PRVOU.SEMT.CRD00604.SEQ | N/A | Perform validations and deliver an output file to IL |
|  | SORT | CRDP.PD.PRVOU.PERP.CRD00101.SEQ | CRDP.PD.PRVOU.SEMT.CRD00605.SEQ | Sort input file to only get header |
|  | DSNTIAUL | N/A | CRDP.PD.PRVOU.SEMT.CRD00606.SEQ | Unload CRDT.PREPAID\_REQ\_ISS for output’s details |
|  | SORT | CRDP.PD.PRVOU.PERP.CRD00101.SEQ | CRDP.PD.PRVOU.SEMT.CRD00609.SEQ | Sort input file to only get details |
|  | SORT | CRDP.PD.PRVOU.SEMT.CRD00609.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00606.SEQ | CRDP.PD.PRVOU.SEMT.CRD00607.SEQ | Join files to retrieve greek names |
|  | SORT | CRDP.PD.PRVOU.SEMT.CRD00605.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00607.SEQ | CRDP.PD.PRVOU.FTOP.CRD00608.SEQ | Append header and details for output |
|  | K86CP | N/A | N/A | Table Archive |
|  | IEFBR14 | CRDP.PD.PRVOU.SEMT.CRD00401.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00402.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00503.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00604.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00605.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00606.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00607.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00609.SEQ | N/A | Deallocation of working files |

#### [KRPPD007] Voucher Massive Re-issuance (Req) use case: Merge files, Validate and Enrich file

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD007** | SORT | CRDP.PD.PRVOU.INTT.K94CPOUT.SEQ | CRDP.PD.PRVOU.SEMT.CRD00707.SEQ | Extract header to file, with detail’s length |
|  | SORT | CRDP.PD.PRVOU.INTT.K94CPOUT.SEQ | CRDP.PD.PRVOU.SEMT.CRD00708.SEQ | Format detail records for new input file |
|  | SORT | CRDP.PD.PRVOU.SEMT.CRD00707.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00708.SEQ | CRDP.PD.PRVOU.INTT.CRD00700.SEQ | Append header and detail for output |
|  | SORT | CRDP.PD.PRVOU.INTT.CRD00700.SEQ | CRDP.PD.PRVOU.SEMT.CRD00702.SEQ | Extract header from input file |
|  | SORT | CRDP.PD.PRVOU.INTT.CRD00700.SEQ | CRDP.PD.PRVOU.SEMT.CRD00703.SEQ | Extract IBANS from input file |
|  | SORT | CRDP.PD.PRVOU.SEMT.CRD00703.SEQ | CRDP.PD.PRVOU.SEMT.CRD00704.SEQ | Remove duplicate IBANS from previous step |
|  | K79CP | CRDP.PD.PRVOU.SEMT.CRD00702.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00704.SEQ | CRDP.PD.PRVOU.SEMT.CRD00705.SEQ | Merge files |
|  | K80CP | CRDP.PD.PRVOU.SEMT.CRD00705.SEQ | CRDP.PD.PRVOU.INTT.CRD00706.SEQ | Perform validations, enrich file with INST\_DEPOSIT\_ACCOUNT, RC, RC\_TXT |

#### [KRPPD008] Voucher Massive Re-issuance (Req) use case: Calculate Card Product Fees

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD008** | SORT | CRDP.PD.PRVOU.INTT.CRD00700.SEQ | CRDP.PD.PRVOU.SEMT.CRD00807.SEQ | Extract details from input file |
|  | SORT | CRDP.PD.PRVOU.INTT.CRD00706.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00807.SEQ | CRDP.PD.PRVOU.SEMT.CRD00808.SEQ | Join files |
|  | K81CP | CRDP.PD.PRVOU.SEMT.CRD00808.SEQ | CRDP.PD.PRVOU.INTT.CRD00809.SEQ | Calculate card production fee for each valid request |

#### [KRPPD009] Voucher Massive Re-issuance (Req) use case: Place holds

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD009** | DSNTIAUL | N/A | CRDP.PD.PRVOU.SEMT.CRD00910.SEQ | Unload CRDT.PREPAID\_REQ\_ISS for sum amounts |
|  | K74CP | CRDP.PD.PRVOU.SEMT.CRD00910.SEQ | N/A | Place holds based on file with amount, and total fee per IBAN |
|  | DSNTIAUL | N/A | CRDP.PD.PRVOU.SEMT.CRD00911.SEQ | Unload CRDT.PREPAID\_REQ\_ISS for SSIS |
|  | SORT | CRDP.PD.PRVOU.SEMT.CRD00911.SEQ | CRDP.PD.PRVOU.FTOP.CRD00912.SEQ | Copy to permanent FTP file and enrich it with header and trailer |
|  | IEFBR14 | CRDP.PD.PRVOU.SEMT.CRD00702.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00703.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00704.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00705.SEQ,  CRDP.PD.PRVOU.INTT.CRD00706.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00807.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00808.SEQ,  CRDP.PD.PRVOU.INTT.CRD00809.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00910.SEQ,  CRDP.PD.PRVOU.SEMT.CRD00911.SEQ | N/A | Deallocation of working files |

#### [KRPPD010] Voucher Massive Re-issuance (Res) use case: Connect SYDIPEL & insert to SMARTPIN

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD010** | SORT | CRDP.PD.PRVOU.FTIP.CRD01000.SEQ | CRDP.PD.PRVOU.SEMT.CRD01001.SEQ | Remove header and trailer and REC\_TYPE from input file |
|  | SORT | CRDP.PD.PRVOU.SEMT.CRD01001.SEQ | CRDP.PD.PRVOU.SEMT.CRD01002.SEQ | Remove header and trailer and REC\_TYPE from input file |
|  | K82CP | CRDP.PD.PRVOU.SEMT.CRD01002.SEQ | N/A | Connect to SYDIPEL and insert into SMARTPIN tables |

#### [KRPPD011] Voucher Massive Re-issuance (Res) use case: Deblock and Debit/Credit account

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD011** | DSNTIAUL | N/A | CRDP.PD.PRVOU.SEMT.CRD01103.SEQ | Unload CRDT.PREPAID\_REQ\_ISS for sum amounts |
|  | K77CP | CRDP.PD.PRVOU.SEMT.CRD01103.SEQ | N/A | Deblock and Debit/Credit account |

#### [KRPPD012] Voucher Massive Re-issuance (Res) use case: Validate files, and Archive tables

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD012** | DSNTIAUL | N/A | CRDP.PD.PRVOU.SEMT.CRD01204.SEQ | Unload CRDT.PREPAID\_REQ\_ISS for log |
|  | K78CP | CRDP.PD.PRVOU.SEMT.CRD01204.SEQ | N/A | Perform validations and deliver an output file to IL |
|  | SORT | CRDP.PD.PRVOU.INTT.CRD00700.SEQ | CRDP.PD.PRVOU.SEMT.CRD01205.SEQ | Extract header from input file |
|  | DSNTIAUL | N/A | CRDP.PD.PRVOU.SEMT.CRD01206.SEQ | Unload CRDT.PREPAID\_REQ\_ISS for output’s details |
|  | SORT | CRDP.PD.PRVOU.SEMT.CRD01205.SEQ,  CRDP.PD.PRVOU.SEMT.CRD01206.SEQ | CRDP.PD.PRVOU.FTOP.CRD01207.SEQ | Append header and detail for output |
|  | K86CP | N/A | N/A | Table archiving |
|  | IEFBR14 | CRDP.PD.PRVOU.SEMT.CRD01001.SEQ,  CRDP.PD.PRVOU.SEMT.CRD01002.SEQ,  CRDP.PD.PRVOU.SEMT.CRD01103.SEQ,  CRDP.PD.PRVOU.SEMT.CRD01204.SEQ,  CRDP.PD.PRVOU.SEMT.CRD01205.SEQ,  CRDP.PD.PRVOU.SEMT.CRD01206.SEQ |  | Deallocation of working files |

#### [KD5DP] Validate file header

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | KD5DPIN: mass issue FTP file  DCL 1 RKD5DPIN\_FH UNALIGNED BASED (ADDR(RKD5DPIN)),  5 HDATE CHAR(10) INIT(' '),  5 HCOUNT PIC'(07)9' INIT(0),  5 SEQ\_NUM PIC'(03)9' INIT(0),  5 TYPE CHAR(02) INIT(' '),  5 ORG\_CD CHAR(03) INIT(' '),  5 HFILLER1 CHAR(267) INIT(' ');  DCL 1 RKD5DPIN\_DT UNALIGNED BASED (ADDR(RKD5DPIN)),  5 LAST\_NAME CHAR(40) INIT(' '),  5 FIRST\_NAME CHAR(40) INIT(' '),  5 FATHER\_NAME CHAR(40) INIT(' '),  5 BIRTH\_DATE CHAR(10) INIT(' '),  5 ID\_TYPE CHAR(02) INIT(' '),  5 ID\_NO CHAR(15) INIT(' '),  5 ID\_ISS\_DATE CHAR(10) INIT(' '),  5 ID\_EXP\_DATE CHAR(10) INIT(' '),  5 ID\_ISS\_CNTRY CHAR(02) INIT(' '),  5 CITIZENSHIP CHAR(02) INIT(' '),  5 TAX\_ID\_NO CHAR(09) INIT(' '),  5 EMPLOEE\_ID CHAR(16) INIT(' '),  5 PE\_IBAN CHAR(27) INIT(' '),  5 DEPARTMENT\_ID CHAR(10) INIT(' '),  5 LE\_IBAN CHAR(27) INIT(' '),  5 LE\_SYDIPEL PIC'(11)9' INIT(0),  5 PE\_SYDIPEL PIC'(11)9' INIT(0),  5 PE\_MOBILE\_PHONE PIC'(10)9' INIT(0); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | KD5DPOUT: mass issue work file  DCL 1 RKD5DPOUT\_FH UNALIGNED BASED (ADDR(RKD5DPOUT)),  5 HDATE CHAR(10) INIT(' '),  5 HCOUNT PIC'(07)9' INIT(0),  5 SEQ\_NUM PIC'(03)9' INIT(0),  5 TYPE CHAR(02) INIT(' '),  5 ORG\_CD CHAR(03) INIT(' '),  5 HFILLER1 CHAR(536) INIT(' ');    DCL 1 RKD5DPOUT\_DT UNALIGNED BASED (ADDR(RKD5DPOUT)),  5 INPUT\_FIELDS CHAR(292) INIT(' '),  5 COMPANY\_NAME CHAR(27) INIT(' '),  5 COMPANY\_ADDR1 CHAR(40) INIT(' '),  5 COMPANY\_ADDR2 CHAR(40) INIT(' '),  5 COMPANY\_ADDR3 CHAR(40) INIT(' '),  5 COMPANY\_ADDR4 CHAR(40) INIT(' '),  5 RC CHAR(02) INIT(' '),  5 RC\_TXT CHAR(80) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_KD5DPIN: Raised when end of file is reached |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve current date.  Check if input file is empty. Check header format. Check file header.  Count records in input file, and compare with count in header.  If no error has appeared so far, transport records from input file to output file: (If there is header but no detail, create detail.)  If at any point an error appears, don’t continue. |
| **PROGRAM MAIN LOOP** | N/A |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.PREPAID\_REQ |

#### [KD5LP] Validate file detail records

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | KD5LPIN: Mass Issue file  DCL 1 RKD5LPIN\_FH UNALIGNED BASED (ADDR(RKD5LPIN)),  5 HDATE CHAR(10) INIT(' '),  5 HCOUNT PIC'(07)9' INIT(0),  5 SEQ\_NUM PIC'(03)9' INIT(0),  5 TYPE CHAR(02) INIT(' '),  5 ORG\_CD CHAR(03) INIT(' '),  5 HFILLER1 CHAR(536) INIT(' ');    DCL 1 RKD5LPIN\_DT UNALIGNED BASED (ADDR(RKD5LPIN)),  5 LAST\_NAME CHAR(40) INIT(' '),  5 FIRST\_NAME CHAR(40) INIT(' '),  5 FATHER\_NAME CHAR(40) INIT(' '),  5 BIRTH\_DATE CHAR(10) INIT(' '),  5 ID\_TYPE CHAR(02) INIT(' '),  5 ID\_NO CHAR(15) INIT(' '),  5 ID\_ISS\_DATE CHAR(10) INIT(' '),  5 ID\_EXP\_DATE CHAR(10) INIT(' '),  5 ID\_ISS\_CNTRY CHAR(02) INIT(' '),  5 CITIZENSHIP CHAR(02) INIT(' '),  5 TAX\_ID\_NO CHAR(09) INIT(' '),  5 EMPLOEE\_ID CHAR(16) INIT(' '),  5 PE\_IBAN CHAR(27) INIT(' '),  5 DEPARTMENT\_ID CHAR(10) INIT(' '),  5 LE\_IBAN CHAR(27) INIT(' '),  5 LE\_SYDIPEL PIC'(11)9' INIT(0),  5 PE\_SYDIPEL PIC'(11)9' INIT(0),  5 PE\_MOBILE\_PHONE PIC'(10)9' INIT(0),  5 COMPANY\_NAME CHAR(27) INIT(' '),  5 COMPANY\_ADDR1 CHAR(40) INIT(' '),  5 COMPANY\_ADDR2 CHAR(40) INIT(' '),  5 COMPANY\_ADDR3 CHAR(40) INIT(' '),  5 COMPANY\_ADDR4 CHAR(40) INIT(' '),  5 RC CHAR(02) INIT(' '),  5 RC\_TXT CHAR(80) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | KD5LPOUT: Mass issue response  DCL 1 RKD5LPOUT,  5 LAST\_NAME CHAR(40) INIT(' '),  5 FIRST\_NAME CHAR(40) INIT(' '),  5 FATHER\_NAME CHAR(40) INIT(' '),  5 BIRTH\_DATE CHAR(10) INIT(' '),  5 ID\_TYPE CHAR(02) INIT(' '),  5 ID\_NO CHAR(15) INIT(' '),  5 ID\_ISS\_DATE CHAR(10) INIT(' '),  5 ID\_EXP\_DATE CHAR(10) INIT(' '),  5 ID\_ISS\_CNTRY CHAR(02) INIT(' '),  5 CITIZENSHIP CHAR(02) INIT(' '),  5 TAX\_ID\_NO CHAR(09) INIT(' '),  5 EMPLOEE\_ID CHAR(16) INIT(' '),  5 PE\_IBAN CHAR(27) INIT(' '),  5 DEPARTMENT\_ID CHAR(10) INIT(' '),  5 LE\_IBAN CHAR(27) INIT(' '),  5 LE\_SYDIPEL PIC'ZZZZZZZZZZ9' INIT(0),  5 PE\_SYDIPEL PIC'(11)9' INIT(0),  5 PE\_MOBILE\_PHONE PIC'(10)9' INIT(0),  5 COMPANY\_NAME CHAR(27) INIT(' '),  5 COMPANY\_ADDR1 CHAR(40) INIT(' '),  5 COMPANY\_ADDR2 CHAR(40) INIT(' '),  5 COMPANY\_ADDR3 CHAR(40) INIT(' '),  5 COMPANY\_ADDR4 CHAR(40) INIT(' '),  5 RC CHAR(02) INIT(' '),  5 RC\_TXT CHAR(80) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_KD5LPIN: Raised when end of file is reached |
| **SERVICES** | SELXWR: Country codes  IBANCHK: Validate IBAN |
| **Main Process Logic** | Retrieve current date.  Retrieve header data.  For NBG files (WSP\_ORG\_CD = 'ETE'), do not run validations. For any other case verify formatting for every record.  Write header to output file. Mark all records with error. Iterate records and write to output file. |
| **PROGRAM MAIN LOOP** | Verify formatting for every record (in non ETE files). |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

#### [KD5RP] Enrich file with company data

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | KD5RPIN: Mass issue file  DCL 1 RKD5RPIN\_FH UNALIGNED BASED (ADDR(RKD5RPIN)),  5 HDATE CHAR(10) INIT(' '),  5 HCOUNT PIC'(07)9' INIT(0),  5 SEQ\_NUM PIC'(03)9' INIT(0),  5 TYPE CHAR(02) INIT(' '),  5 ORG\_CD CHAR(03) INIT(' '),  5 HFILLER1 CHAR(536) INIT(' ');    DCL 1 RKD5RPIN\_DT UNALIGNED BASED (ADDR(RKD5RPIN)),  5 LAST\_NAME CHAR(40) INIT(' '),  5 FIRST\_NAME CHAR(40) INIT(' '),  5 FATHER\_NAME CHAR(40) INIT(' '),  5 BIRTH\_DATE CHAR(10) INIT(' '),  5 ID\_TYPE CHAR(02) INIT(' '),  5 ID\_NO CHAR(15) INIT(' '),  5 ID\_ISS\_DATE CHAR(10) INIT(' '),  5 ID\_EXP\_DATE CHAR(10) INIT(' '),  5 ID\_ISS\_CNTRY CHAR(02) INIT(' '),  5 CITIZENSHIP CHAR(02) INIT(' '),  5 TAX\_ID\_NO CHAR(09) INIT(' '),  5 EMPLOEE\_ID CHAR(16) INIT(' '),  5 PE\_IBAN CHAR(27) INIT(' '),  5 DEPARTMENT\_ID CHAR(10) INIT(' '),  5 LE\_IBAN CHAR(27) INIT(' '),  5 LE\_SYDIPEL PIC'(11)9' INIT(0),  5 PE\_SYDIPEL PIC'(11)9' INIT(0),  5 PE\_MOBILE\_PHONE PIC'(10)9' INIT(0),  5 COMPANY\_NAME CHAR(27) INIT(' '),  5 COMPANY\_ADDR1 CHAR(40) INIT(' '),  5 COMPANY\_ADDR2 CHAR(40) INIT(' '),  5 COMPANY\_ADDR3 CHAR(40) INIT(' '),  5 COMPANY\_ADDR4 CHAR(40) INIT(' '),  5 RC CHAR(02) INIT(' '),  5 RC\_TXT CHAR(80) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | KD5RPOUT: Mass issue response  DCL 1 RKD5RPOUT,  5 LAST\_NAME CHAR(40) INIT(' '),  5 FIRST\_NAME CHAR(40) INIT(' '),  5 FATHER\_NAME CHAR(40) INIT(' '),  5 BIRTH\_DATE CHAR(10) INIT(' '),  5 ID\_TYPE CHAR(02) INIT(' '),  5 ID\_NO CHAR(15) INIT(' '),  5 ID\_ISS\_DATE CHAR(10) INIT(' '),  5 ID\_EXP\_DATE CHAR(10) INIT(' '),  5 ID\_ISS\_CNTRY CHAR(02) INIT(' '),  5 CITIZENSHIP CHAR(02) INIT(' '),  5 TAX\_ID\_NO CHAR(09) INIT(' '),  5 EMPLOEE\_ID CHAR(16) INIT(' '),  5 PE\_IBAN CHAR(27) INIT(' '),  5 DEPARTMENT\_ID CHAR(10) INIT(' '),  5 LE\_IBAN CHAR(27) INIT(' '),  5 LE\_SYDIPEL PIC'ZZZZZZZZZZ9' INIT(0),  5 PE\_SYDIPEL PIC'(11)9' INIT(0),  5 PE\_MOBILE\_PHONE PIC'(10)9' INIT(0),  5 COMPANY\_NAME CHAR(27) INIT(' '),  5 COMPANY\_ADDR1 CHAR(40) INIT(' '),  5 COMPANY\_ADDR2 CHAR(40) INIT(' '),  5 COMPANY\_ADDR3 CHAR(40) INIT(' '),  5 COMPANY\_ADDR4 CHAR(40) INIT(' '),  5 RC CHAR(02) INIT(' '),  5 RC\_TXT CHAR(80) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_KD5RPIN: Raised when end of file is reached |
| **SERVICES** | SRVIIPS: Customer information  SRVINMF: Fetch names  SRVIADR: Fetch customer address  SRVIPDF: Fetch customers from CRAT.C\_IP\_DOC  SRVIXAA: Fetch from CRAT.C\_IP\_X\_ACC by account  SRVIAIH: Fetch from CRAT.C\_IP\_ACC\_INDC and C\_IP\_ACC\_INDC\_HIST by SYDIPEL |
| **Main Process Logic** | Retrieve current date. Retrieve date from file header.  If LE\_IBAN and LE\_SYDIPEL have been validated successfully (RKD5RPIN\_DT.RC != 31/37/32) then perform legal entity validations from the first detail record. Otherwise, everything is wrong, and cannot be validated further.  Mark every record form input file with error response codes.  Write detail records from input file to output file. |
| **PROGRAM MAIN LOOP** | N/A |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

#### [KD6NP] Update file records

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | KD6NPDP: Duplicates file  DCL 1 RKD6NPDP CHAR(561) INIT(' ');    DCL 1 RKD6NPDP\_DT UNALIGNED BASED (ADDR(RKD6NPDP)),  5 LAST\_NAME CHAR(40) INIT(' '),  5 FIRST\_NAME CHAR(40) INIT(' '),  5 FATHER\_NAME CHAR(40) INIT(' '),  5 BIRTH\_DATE CHAR(10) INIT(' '),  5 ID\_TYPE CHAR(02) INIT(' '),  5 ID\_NO CHAR(15) INIT(' '),  5 ID\_ISS\_DATE CHAR(10) INIT(' '),  5 ID\_EXP\_DATE CHAR(10) INIT(' '),  5 ID\_ISS\_CNTRY CHAR(02) INIT(' '),  5 CITIZENSHIP CHAR(02) INIT(' '),  5 TAX\_ID\_NO CHAR(09) INIT(' '),  5 EMPLOEE\_ID CHAR(16) INIT(' '),  5 PE\_IBAN CHAR(27) INIT(' '),  5 DEPARTMENT\_ID CHAR(10) INIT(' '),  5 LE\_IBAN CHAR(27) INIT(' '),  5 LE\_SYDIPEL PIC'(11)9' INIT(0),  5 PE\_SYDIPEL PIC'(11)9' INIT(0),  5 PE\_MOBILE\_PHONE PIC'(10)9' INIT(0),  5 COMPANY\_NAME CHAR(27) INIT(' '),  5 COMPANY\_ADDR1 CHAR(40) INIT(' '),  5 COMPANY\_ADDR2 CHAR(40) INIT(' '),  5 COMPANY\_ADDR3 CHAR(40) INIT(' '),  5 COMPANY\_ADDR4 CHAR(40) INIT(' '),  5 RC CHAR(02) INIT(' '),  5 RC\_TXT CHAR(80) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **UPDATE1 FILE DECLARATION** | KD6NPIN: Mass issue file  DCL 1 RKD6NPIN CHAR(561) INIT(' ');    DCL 1 RKD6NPIN\_FH UNALIGNED BASED (ADDR(RKD6NPIN)),  5 HDATE CHAR(10) INIT(' '),  5 HCOUNT PIC'(07)9' INIT(0),  5 SEQ\_NUM PIC'(03)9' INIT(0),  5 TYPE CHAR(02) INIT(' '),  5 ORG\_CD CHAR(03) INIT(' '),  5 HFILLER1 CHAR(536) INIT(' ');  DCL 1 RKD6NPIN\_DT UNALIGNED BASED (ADDR(RKD6NPIN)),  5 LAST\_NAME CHAR(40) INIT(' '),  5 FIRST\_NAME CHAR(40) INIT(' '),  5 FATHER\_NAME CHAR(40) INIT(' '),  5 BIRTH\_DATE CHAR(10) INIT(' '),  5 ID\_TYPE CHAR(02) INIT(' '),  5 ID\_NO CHAR(15) INIT(' '),  5 ID\_ISS\_DATE CHAR(10) INIT(' '),  5 ID\_EXP\_DATE CHAR(10) INIT(' '),  5 ID\_ISS\_CNTRY CHAR(02) INIT(' '),  5 CITIZENSHIP CHAR(02) INIT(' '),  5 TAX\_ID\_NO CHAR(09) INIT(' '),  5 EMPLOEE\_ID CHAR(16) INIT(' '),  5 PE\_IBAN CHAR(27) INIT(' '),  5 DEPARTMENT\_ID CHAR(10) INIT(' '),  5 LE\_IBAN CHAR(27) INIT(' '),  5 LE\_SYDIPEL PIC'(11)9' INIT(0),  5 PE\_SYDIPEL PIC'(11)9' INIT(0),  5 PE\_MOBILE\_PHONE PIC'(10)9' INIT(0),  5 COMPANY\_NAME CHAR(27) INIT(' '),  5 COMPANY\_ADDR1 CHAR(40) INIT(' '),  5 COMPANY\_ADDR2 CHAR(40) INIT(' '),  5 COMPANY\_ADDR3 CHAR(40) INIT(' '),  5 COMPANY\_ADDR4 CHAR(40) INIT(' '),  5 RC CHAR(02) INIT(' '),  5 RC\_TXT CHAR(80) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_KD6NPIN: Raised when end of file is reached |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve current date.  Read through records in KD6NPIN: For RC 00, AFM isn’t a duplicate. Update record. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of file KD6NPIN:   * **If** record response code is 00 **then** set record response code to 13, and response text to non-duplicate AFM * Rewrite file for record |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

#### [KD6OP] Translate greek names

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | KD6OPIN: Mass issue file  DCL 1 RKD6OPIN CHAR(561) INIT(' ');    DCL 1 RKD6OPIN\_FH UNALIGNED BASED (ADDR(RKD6OPIN)),  5 HDATE CHAR(10) INIT(' '),  5 HCOUNT PIC'(07)9' INIT(0),  5 SEQ\_NUM PIC'(03)9' INIT(0),  5 TYPE CHAR(02) INIT(' '),  5 ORG\_CD CHAR(03) INIT(' '),  5 HFILLER1 CHAR(536) INIT(' ');  DCL 1 RKD6OPIN\_DT UNALIGNED BASED (ADDR(RKD6OPIN)),  5 LAST\_NAME CHAR(40) INIT(' '),  5 FIRST\_NAME CHAR(40) INIT(' '),  5 FATHER\_NAME CHAR(40) INIT(' '),  5 BIRTH\_DATE CHAR(10) INIT(' '),  5 ID\_TYPE CHAR(02) INIT(' '),  5 ID\_NO CHAR(15) INIT(' '),  5 ID\_ISS\_DATE CHAR(10) INIT(' '),  5 ID\_EXP\_DATE CHAR(10) INIT(' '),  5 ID\_ISS\_CNTRY CHAR(02) INIT(' '),  5 CITIZENSHIP CHAR(02) INIT(' '),  5 TAX\_ID\_NO CHAR(09) INIT(' '),  5 EMPLOEE\_ID CHAR(16) INIT(' '),  5 PE\_IBAN CHAR(27) INIT(' '),  5 DEPARTMENT\_ID CHAR(10) INIT(' '),  5 LE\_IBAN CHAR(27) INIT(' '),  5 LE\_SYDIPEL PIC'(11)9' INIT(0),  5 PE\_SYDIPEL PIC'(11)9' INIT(0),  5 PE\_MOBILE\_PHONE PIC'(10)9' INIT(0),  5 COMPANY\_NAME CHAR(27) INIT(' '),  5 COMPANY\_ADDR1 CHAR(40) INIT(' '),  5 COMPANY\_ADDR2 CHAR(40) INIT(' '),  5 COMPANY\_ADDR3 CHAR(40) INIT(' '),  5 COMPANY\_ADDR4 CHAR(40) INIT(' '),  5 RC CHAR(02) INIT(' '),  5 RC\_TXT CHAR(80) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | KD6OPOUT: SEMT with Latin  DCL 1 RKD6OPOUT,  5 LAST\_NAME CHAR(40) INIT(' '),  5 FIRST\_NAME CHAR(40) INIT(' '),  5 FATHER\_NAME CHAR(40) INIT(' '),  5 BIRTH\_DATE CHAR(10) INIT(' '),  5 ID\_TYPE CHAR(02) INIT(' '),  5 ID\_NO CHAR(15) INIT(' '),  5 ID\_ISS\_DATE CHAR(10) INIT(' '),  5 ID\_EXP\_DATE CHAR(10) INIT(' '),  5 ID\_ISS\_CNTRY CHAR(02) INIT(' '),  5 CITIZENSHIP CHAR(02) INIT(' '),  5 TAX\_ID\_NO CHAR(09) INIT(' '),  5 EMPLOEE\_ID CHAR(16) INIT(' '),  5 PE\_IBAN CHAR(27) INIT(' '),  5 DEPARTMENT\_ID CHAR(10) INIT(' '),  5 LE\_IBAN CHAR(27) INIT(' '),  5 LE\_SYDIPEL PIC'ZZZZZZZZZZ9' INIT(0),  5 PE\_SYDIPEL PIC'(11)9' INIT(0),  5 PE\_MOBILE\_PHONE PIC'(10)9' INIT(0),  5 COMPANY\_NAME CHAR(27) INIT(' '),  5 COMPANY\_ADDR1 CHAR(40) INIT(' '),  5 COMPANY\_ADDR2 CHAR(40) INIT(' '),  5 COMPANY\_ADDR3 CHAR(40) INIT(' '),  5 COMPANY\_ADDR4 CHAR(40) INIT(' '),  5 RC CHAR(02) INIT(' '),  5 RC\_TXT CHAR(80) INIT(' '),  5 LAST\_NAME\_LAT CHAR(40) INIT(' '),  5 FIRST\_NAME\_LAT CHAR(40) INIT(' '),  5 FATHER\_NAME\_LAT CHAR(40) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_KD6OPIN: Raised when end of file is reached |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve current date.  Write header from input file to output file.  Read details records from input file: Translate name, and write records to output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * Translate last name, first name, and father name * Write record to output file |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

#### [KD6PP] Check suspect

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | IFILE:  DCL RIFILE CHAR(681) ;  DCL 1 DRIFILE UNALIGNED BASED (ADDR(RIFILE)),  5 LAST\_NAME CHAR(40) INIT(' '),  5 FIRST\_NAME CHAR(40) INIT(' '),  5 FATHER\_NAME CHAR(40) INIT(' '),  5 BIRTH\_DATE CHAR(10) INIT(' '),  5 ID\_TYPE CHAR(02) INIT(' '),  5 ID\_NO CHAR(15) INIT(' '),  5 ID\_ISS\_DATE CHAR(10) INIT(' '),  5 ID\_EXP\_DATE CHAR(10) INIT(' '),  5 ID\_ISS\_CNTRY CHAR(02) INIT(' '),  5 CITIZENSHIP CHAR(02) INIT(' '),  5 TAX\_ID\_NO CHAR(09) INIT(' '),  5 EMPLOEE\_ID CHAR(16) INIT(' '),  5 PE\_IBAN CHAR(27) INIT(' '),  5 DEPARTMENT\_ID CHAR(10) INIT(' '),  5 LE\_IBAN CHAR(27) INIT(' '),  5 LE\_SYDIPEL PIC'ZZZZZZZZZZ9' INIT(0),  5 PE\_SYDIPEL PIC'(11)9' INIT(0),  5 PE\_MOBILE\_PHONE PIC'(10)9' INIT(0),  5 COMPANY\_NAME CHAR(27) INIT(' '),  5 COMPANY\_ADDR1 CHAR(40) INIT(' '),  5 COMPANY\_ADDR2 CHAR(40) INIT(' '),  5 COMPANY\_ADDR3 CHAR(40) INIT(' '),  5 COMPANY\_ADDR4 CHAR(40) INIT(' '),  5 RC CHAR(02) INIT(' '),  5 RC\_TXT CHAR(80) INIT(' '),  5 LAST\_NAME\_LAT CHAR(40) INIT(' '),  5 FIRST\_NAME\_LAT CHAR(40) INIT(' '),  5 FATHER\_NAME\_LAT CHAR(40) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | KD6PPOUT: SEMT  DCL 1 RKD6PPOUT,  5 LAST\_NAME CHAR(40) INIT(' '),  5 FIRST\_NAME CHAR(40) INIT(' '),  5 FATHER\_NAME CHAR(40) INIT(' '),  5 BIRTH\_DATE CHAR(10) INIT(' '),  5 ID\_TYPE CHAR(02) INIT(' '),  5 ID\_NO CHAR(15) INIT(' '),  5 ID\_ISS\_DATE CHAR(10) INIT(' '),  5 ID\_EXP\_DATE CHAR(10) INIT(' '),  5 ID\_ISS\_CNTRY CHAR(02) INIT(' '),  5 CITIZENSHIP CHAR(02) INIT(' '),  5 TAX\_ID\_NO CHAR(09) INIT(' '),  5 EMPLOEE\_ID CHAR(16) INIT(' '),  5 PE\_IBAN CHAR(27) INIT(' '),  5 DEPARTMENT\_ID CHAR(10) INIT(' '),  5 LE\_IBAN CHAR(27) INIT(' '),  5 LE\_SYDIPEL PIC'ZZZZZZZZZZ9' INIT(0),  5 PE\_SYDIPEL PIC'(11)9' INIT(0),  5 PE\_MOBILE\_PHONE PIC'(10)9' INIT(0),  5 COMPANY\_NAME CHAR(27) INIT(' '),  5 COMPANY\_ADDR1 CHAR(40) INIT(' '),  5 COMPANY\_ADDR2 CHAR(40) INIT(' '),  5 COMPANY\_ADDR3 CHAR(40) INIT(' '),  5 COMPANY\_ADDR4 CHAR(40) INIT(' '),  5 RC CHAR(02) INIT(' '),  5 RC\_TXT CHAR(80) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE: Raised when end of file is reached |
| **SERVICES** | N/A |
| **Main Process Logic** | Fetch suspect names, and for every one: Parse names, and process them. Log counter every 1000 processes.  Write header to output file.  Read details records from input file: Check is customer needs to visit an NBG store. Write data to output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * For response code 00:   + Make name and parse it   + If needed, set response code to 40, and text to display need to visit an NBG store   + Write data to output file |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | DXPT.SUSPECT\_NAMES |

#### [KD5SP] Enrich file with owner data

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | FILE1I:  DCL 1 RECIN,  2 LAST\_NAME CHAR(40) INIT(' '),  2 FIRST\_NAME CHAR(40) INIT(' '),  2 FATHER\_NAME CHAR(40) INIT(' '),  2 BIRTH\_DATE CHAR(10) INIT(' '),  2 ID\_TYPE CHAR(02) INIT(' '),  2 ID\_NO CHAR(15) INIT(' '),  2 ID\_ISS\_DATE CHAR(10) INIT(' '),  2 ID\_EXP\_DATE CHAR(10) INIT(' '),  2 ID\_ISS\_CTRY CHAR(02) INIT(' '),  2 CITIZENSHIP CHAR(02) INIT(' '),  2 TAX\_ID\_NO CHAR(09) INIT(' '),  2 EMPLOYEE\_ID CHAR(16) INIT(' '),  2 PE\_IBAN CHAR(27) INIT(' '),  2 DEPARTMENT\_ID CHAR(10) INIT(' '),  2 LE\_IBAN CHAR(27) INIT(' '),  2 LE\_SIDIPEL CHAR(11) INIT(' '),  2 PE\_SIDIPEL CHAR(11) INIT(' '),  2 PE\_MOBILE\_PHONE CHAR(10) INIT(' '),  2 COMPANY\_NAME CHAR(27) INIT(' '),  2 COMPANY\_ADDRESS\_1 CHAR(40) INIT(' '),  2 COMPANY\_ADDRESS\_2 CHAR(40) INIT(' '),  2 COMPANY\_ADDRESS\_3 CHAR(40) INIT(' '),  2 COMPANY\_ADDRESS\_4 CHAR(40) INIT(' '),  2 RC CHAR(02) INIT(' '),  2 RC\_TXT CHAR(80) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | FILE1O:  DCL 1 RECOUT,  2 LAST\_NAME CHAR(40) INIT(' '),  2 FIRST\_NAME CHAR(40) INIT(' '),  2 FATHER\_NAME CHAR(40) INIT(' '),  2 BIRTH\_DATE CHAR(10) INIT(' '),  2 ID\_TYPE CHAR(02) INIT(' '),  2 ID\_NO CHAR(15) INIT(' '),  2 ID\_ISS\_DATE CHAR(10) INIT(' '),  2 ID\_EXP\_DATE CHAR(10) INIT(' '),  2 ID\_ISS\_CTRY CHAR(02) INIT(' '),  2 CITIZENSHIP CHAR(02) INIT(' '),  2 TAX\_ID\_NO CHAR(09) INIT(' '),  2 EMPLOYEE\_ID CHAR(16) INIT(' '),  2 PE\_IBAN CHAR(27) INIT(' '),  2 DEPARTMENT\_ID CHAR(10) INIT(' '),  2 LE\_IBAN CHAR(27) INIT(' '),  2 LE\_SIDIPEL CHAR(11) INIT(' '),  2 PE\_SIDIPEL CHAR(11) INIT(' '),  2 PE\_MOBILE\_PHONE CHAR(10) INIT(' '),  2 COMPANY\_NAME CHAR(27) INIT(' '),  2 COMPANY\_ADDRESS\_1 CHAR(40) INIT(' '),  2 COMPANY\_ADDRESS\_2 CHAR(40) INIT(' '),  2 COMPANY\_ADDRESS\_3 CHAR(40) INIT(' '),  2 COMPANY\_ADDRESS\_4 CHAR(40) INIT(' '),  2 BR\_ACT CHAR(01) INIT(' '),  2 RC CHAR(02) INIT(' '),  2 RC\_TXT CHAR(80) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | SDATA: Raised when end of file is reached |
| **SERVICES** | INDSEL1: Select from CRAT.C\_IP\_INDC  CGESEL3: Select from CRAT.C\_GEO  CIPSEL: Select from CRAT.C\_IP  IAPFET1: Select from CRAT.C\_IP\_X\_APPL  CLSSEL: Select from CRAT.C\_CLSF  SRVCLAT: Select from CRAT.C\_LOG\_ADDR |
| **Main Process Logic** | Copy header from input file to output file.  Read details records from input file: For non rejected records, validate AFM, last name, first name, and father name, and find customer and phone. Write data to output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * **If** the record has not been rejected (response code 00) **then**   + **If** AFM (tax id number) is empty, **then** set response code to 33, and text to show field as mandatory   + **Else if** last name is empty or includes invalid characters, **then** set response code to 20, and text to show field as mandatory with no special characters, and write data to output file   + **Else if** firstname is empty or includes invalid characters, **then** set response code to 21, and text to show field as mandatory with no special characters, and write data to output file   + **Else if** father name is empty or includes invalid characters, **then** set response code to 22, and text to show field as mandatory with no special characters, and write data to output file   + **Else**      - Set values     - Try to find customer by       * Account and SYDIPEL or       * AFM and Account or       * AFM and SYDIPEL or       * Account and name or       * AFM and name or       * Legal document and name     - **If** customer wasn’t found, **then**       * Insert customer     - **Else** (customer was found)       * **If** transaction is suspicious (by calling INDSEL1 with @INDSEL1.IDATA.I\_CLSF\_INDC\_TP\_K=8800001), **then** set response code to 03, and set text to show need to visit NBG store, and write data to output file       * **Else** fill BR\_ACT field,search for phone (if not found, insert phone), and write data to output file * **Else** (record has already been rejected) write data to output file * SQL commit |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

#### [K79CP] Merging files

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | INPFILE1: Header  DCL 1 I\_RECORD\_1,  5 HEADER CHAR(25) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | INPFILE2: IBAN, SYDIPEL  DCL 1 I\_RECORD\_2,  5 IBAN CHAR(27) INIT(' '),  5 SYDIPEL CHAR(11) INIT(' '),  5 RC CHAR(02) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | OUTFILE1:  DCL 1 O\_RECORD,  5 CONCAT\_FIELD CHAR(65) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_INPUT1: Raised when end of file is reached MORE\_INPUT2: Raised when end of file is reached |
| **SERVICES** | N/A |
| **Main Process Logic** | For every record from INPFILE2: Fill output and write to output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file INPFILE2:   * Write a record to output file as a combination of the input files, header (from INPFILE1) and IBAN, SYDIPEL, RC (from INPFILE2) |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

#### [K80CP] Perform Validations, Enrich file

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | INPFILE1:  DCL 1 I\_RECORD,  5 BUSINESS\_DATE CHAR(10) INIT(' '),  5 TOTAL\_REQS\_CNT PIC'(7)9' INIT(0),  5 SEQUENCE\_NUMBER CHAR(3) INIT(' '),  5 TYPE CHAR(2) INIT(' '),  5 ORGANIZATION\_CODE CHAR(3) INIT(' '),  5 IBAN CHAR(27) INIT(' '),  5 LE\_SYDIPEL PIC'(11)9' INIT(0),  5 RC CHAR(002) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | OUTFILE1:  DCL 1 O\_RECORD,  5 BUSINESS\_DATE CHAR(10) INIT(' '),  5 TOTAL\_REQS\_CNT PIC'(7)9' INIT(0),  5 SEQUENCE\_NUMBER CHAR(3) INIT(' '),  5 TYPE CHAR(2) INIT(' '),  5 ORGANIZATION\_CODE CHAR(3) INIT(' '),  5 IBAN CHAR(27) INIT(' '),  5 INSTITUTION\_D\_ACCOUNT CHAR(16) INIT(' '),  5 LE\_SYDIPEL PIC'(11)9' INIT(0),  5 RC CHAR(002) INIT(' '),  5 RC\_TXT CHAR(80) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_INPUT1: Raised when end of file is reached |
| **SERVICES** | SRVSIBA: IBAN-Account matching  SRVIXAS: Select from CRAT.C\_IP\_X\_ACC, CRAT.C\_IP  SRVCKCS: Retrieve customers connected with card  ICDPCM1: Insert into CRDT\_PREPAID\_REQ |
| **Main Process Logic** | For every record from input file: For non rejected records retrieve current date, institution deposit account, SYDIPEL, validate data and insert data to CRDT.LOADING\_MASTER. Write data to output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file input file:   * For records with response code 00:   + Retrieve current date   + Retrieve institution deposit account (by calling SRVSIBA)   + Retrieve legal SYDIPEL in case of reinsurance     - For record type IS: from input file     - Otherwise by calling SRVIXAS   + For organization code not NBG (I\_RECORD.ORGANIZATION\_CODE!=ETE) validate customer data (by calling SRVCKCS)   + Insert data into CRDT.LOADING\_MASTER (by calling ICDPCM1)   + Commit action * Write data to output file |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.LOADING\_MASTER |

#### [K81CP] Calculate Card Production Fee for each valid request

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | INPFILE1: Merged flat file arrived from card division  DCL 1 I\_RECORD,  5 BUSINESS\_DATE CHAR(10) INIT(' '),  5 SEQ\_NO CHAR(03) INIT(' '),  5 TYPE CHAR(02) INIT(' '),  5 ORG\_CDE CHAR(03) INIT(' '),  5 PRODUCTION\_EVENT CHAR(03) INIT(' '),  5 PAN CHAR(19) INIT(' '),  5 PROD\_CODE CHAR(03) INIT(' '),  5 NEW\_PAN CHAR(19) INIT(' '),  5 NEW\_PRD\_CODE CHAR(03) INIT(' '),  5 CRD\_EXP\_DAT CHAR(10) INIT(' '),  5 PROD\_NAME CHAR(30) INIT(' '),  5 PROD\_GROUP CHAR(01) INIT(' '),  5 SMARTPIN\_IND CHAR(01) INIT(' '),  5 LAST\_NAME CHAR(40) INIT(' '),  5 FIRST\_NAME CHAR(40) INIT(' '),  5 FATHER\_NAME CHAR(40) INIT(' '),  5 BIRTH\_DATE CHAR(10) INIT(' '),  5 ID\_TYPE PIC'(2)9' INIT(0),  5 ID\_NO CHAR(15) INIT(' '),  5 ID\_ISS\_DATE CHAR(10) INIT(' '),  5 ID\_EXP\_DATE CHAR(10) INIT(' '),  5 ID\_ISS\_CTRY CHAR(02) INIT(' '),  5 CITIZENSHIP CHAR(02) INIT(' '),  5 TAX\_ID\_NO CHAR(09) INIT(' '),  5 EMPLOYEE\_ID CHAR(16) INIT(' '),  5 PE\_IBAN CHAR(27) INIT(' '),  5 CARD\_ID CHAR(20) INIT(' '),  5 DEPARTMENT\_ID CHAR(10) INIT(' '),  5 LE\_IBAN CHAR(27) INIT(' '),  5 LE\_DEP\_ACC CHAR(16) INIT(' '),  5 LE\_SYDIPEL PIC'(11)9' INIT(0),  5 PE\_SYDIPEL PIC'(11)9' INIT(0),  5 MOBILE CHAR(10) INIT(0),  5 COMPANY\_NAME CHAR(27) INIT(' '),  5 COMPANY\_ADDRESS\_1 CHAR(40) INIT(' '),  5 COMPANY\_ADDRESS\_2 CHAR(40) INIT(' '),  5 COMPANY\_ADDRESS\_3 CHAR(40) INIT(' '),  5 COMPANY\_ADDRESS\_4 CHAR(40) INIT(' '),  5 PE\_SIDIPEL\_FLAG CHAR(01) INIT(' '),  5 REQUEST\_STATUS CHAR(01) INIT(' '),  5 RC CHAR(02) INIT(' '),  5 REJECTION\_TXT CHAR(80) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | OUTFILE1: Enriched flat file  DCL 1 O\_RECORD,  5 BUSINESS\_DATE CHAR(10) INIT(' '),  5 SEQ\_NO CHAR(03) INIT(' '),  5 TYPE CHAR(02) INIT(' '),  5 ORG\_CDE CHAR(03) INIT(' '),  5 UNIQUE\_ID CHAR(27) INIT(' '),  5 PRODUCTION\_EVENT CHAR(03) INIT(' '),  5 AMOUNT PIC'(13)9V.99' INIT(0),  5 FEE PIC'(13)9V.99' INIT(0),  5 COMM\_LOG\_ID CHAR(26) INIT(' '),  5 PAN CHAR(19) INIT(' '),  5 PROD\_CODE CHAR(03) INIT(' '),  5 NEW\_PAN CHAR(19) INIT(' '),  5 NEW\_PRD\_CODE CHAR(03) INIT(' '),  5 CRD\_EXP\_DAT CHAR(10) INIT(' '),  5 PROD\_NAME CHAR(30) INIT(' '),  5 PROD\_GROUP CHAR(01) INIT(' '),  5 SMARTPIN\_IND CHAR(01) INIT(' '),  5 LAST\_NAME CHAR(40) INIT(' '),  5 FIRST\_NAME CHAR(40) INIT(' '),  5 FATHER\_NAME CHAR(40) INIT(' '),  5 BIRTH\_DATE CHAR(10) INIT(' '),  5 ID\_TYPE CHAR(02) INIT(' '),  5 ID\_NO CHAR(15) INIT(' '),  5 ID\_ISS\_DATE CHAR(10) INIT(' '),  5 ID\_EXP\_DATE CHAR(10) INIT(' '),  5 ID\_ISS\_CTRY CHAR(02) INIT(' '),  5 CITIZENSHIP CHAR(02) INIT(' '),  5 TAX\_ID\_NO CHAR(09) INIT(' '),  5 EMPLOYEE\_ID CHAR(16) INIT(' '),  5 PE\_IBAN CHAR(27) INIT(' '),  5 CARD\_ID CHAR(20) INIT(' '),  5 DEPARTMENT\_ID CHAR(10) INIT(' '),  5 LE\_IBAN CHAR(27) INIT(' '),  5 LE\_DEP\_ACC CHAR(16) INIT(' '),  5 LE\_SYDIPEL PIC'(11)9' INIT(0),  5 PE\_SYDIPEL PIC'(11)9' INIT(0),  5 COMPANY\_NAME CHAR(27) INIT(' '),  5 COMPANY\_ADDRESS\_1 CHAR(40) INIT(' '),  5 COMPANY\_ADDRESS\_2 CHAR(40) INIT(' '),  5 COMPANY\_ADDRESS\_3 CHAR(40) INIT(' '),  5 COMPANY\_ADDRESS\_4 CHAR(40) INIT(' '),  5 REQUEST\_STATUS CHAR(01) INIT(' '),  5 RC CHAR(02) INIT(' '),  5 REJECTION\_TXT CHAR(80) INIT(' '),  5 INS\_TMTSMP CHAR(26) INIT('0001-01-01-01.01.01.000001'),  5 MOBILE CHAR(10) INIT(0),  5 PE\_SIDIPEL\_FLAG CHAR(01) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_INPUT1: Raised when end of file is reached |
| **SERVICES** | ICDISS: Insert into CRDT.PREPAID\_REQ\_ISS  SCCSCAL: Calculate fee requirement  SRVIIPS: Retrieve customer’s information  SRVACMP: Retrieve LE\_SYDIPEL from CRDT.PREPAID\_REQ\_ISS\_HIST  SRVIXAA: Fetch from CRAT.C\_IP\_X\_ACC by account |
| **Main Process Logic** | For every record from input file: Retrieve current date. For non rejected records: translate names, and for record type RE check deposit account, and for request status 0 get customer data and commit action. Write data to output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file input file:   * Retrieve current time * For record response code (RC) ‘00’/’0’/’ ‘/’ ‘:   + Translate names (first, last, father, company)   + For record type RE:     - Retrieve LE\_SYDIPEL from CRDT.PREPAID\_REQ\_ISS\_HIST (by calling SRVACMP)     - Check deposit account (by calling SRVIXAA), and set response code to E8 for account found for a different company   + For request status 0:     - Convert 2digit type to full type and generate prod event     - Generate unique id     - Retrieve customer’s branch (by calling SRVIIPS)     - Retrieve commission amount (by calling SCCSCAL)     - Commit action * Write data to output file |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.PREPAID\_REQ\_ISS  CRDT.PREPAID\_REQ\_ISS\_HIST |

#### 

#### [K74CP] Place hold on accounts

(see section [Massive (Un)Loading Voucher](#_yya6y0h2v9xy))

#### [K82CP] Connect to physical SYDIPEL and update detailed table

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | INPFILE1: Flat file arrived from SSIS that contains the response from WAY4 concerning prepaid card’s issuance/reissuance  DCL 1 I\_RECORD,  5 BUSINESS\_DATE CHAR(10) INIT(' '),  5 SEQ\_NO CHAR(3) INIT(' '),  5 TYPE CHAR(2) INIT(' '),  5 ORG\_CDE CHAR(3) INIT(' '),  5 UNIQUE\_ID CHAR(27) INIT(' '),  5 CARD\_ID CHAR(20) INIT(' '),  5 LE\_SIDIPEL PIC'(11)9' INIT(0),  5 PRODUCTION\_EVENT CHAR(3) INIT(' '),  5 PE\_SIDIPEL PIC'(11)9' INIT(0),  5 PAN CHAR(19) INIT(' '),  5 CRD\_EXP\_DAT CHAR(10) INIT(' '),  5 PROD\_CODE CHAR(3) INIT(' '),  5 PROD\_NAME CHAR(30) INIT(' '),  5 PROD\_GROUP CHAR(1) INIT(' '),  5 SMARTPIN\_IND CHAR(1) INIT(' '),  5 RC CHAR(2) INIT(' '),  5 REJECTION\_TXT CHAR(80) INIT(' '),  5 OLD\_PAN CHAR(19) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_INPUT1: Raised when end of file is reached |
| **SERVICES** | UCDISS2: Update into CRDT.PREPAID\_REQ\_ISS based on UNIQUE\_ID  UCDISS3: Update into CRDT.PREPAID\_REQ\_ISS based on UNIQUE\_ID  UCDISS7: Update into CRDT.PREPAID\_REQ\_ISS based on UNIQUE\_ID  SRVIXAD: Delete rows from CRAT.C\_IP\_X\_ACC  SRVIXAI: Connect Costumer-Account, insert into CRAT.C\_IP\_X\_ACC  ICCSPN: Insert into KCCT.SMARTPIN  ICDSPT: Insert into CRDT.SMARTPIN\_TRACK |
| **Main Process Logic** | For every line of the input file: For non rejected record type IS update CRDT.PREPAID\_REQ\_ISS, establish connections to SYDIPEL, and insert data to KCCT.CARDS\_PIN. For rejected record type IS, update CRDT.PREPAID\_REQ\_ISS accordingly. For non rejected other records update CRDT.PREPAID\_REQ\_ISS, and is PAN will remain the same, no deletion od SYDIPEL is needed, otherwise: delete connection to SYDIPEL, and establish it anew, then delete old record from KCCT.CARDS\_PIN and add a new one. For rejected other records update CRDT.PREPAID\_REQ\_ISS accordingly. For non rejected SMARTPIN PLA action: If PAN is to be changed insert request to KCCT.SMARTPIN, and track to CRDT.SMARTPIN\_TRACK, otherwise update CRDT.PREPAID\_REQ\_ISS. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * Retrieve current date * **For** record type IS:   + For non rejected records (I\_RECORD.RC=00):     - Update CRDT.PREPAID\_REQ\_ISS with the returned data (by calling UCDISS3)     - Establish connection of legal entity/card (by calling SRVIXAI)     - Establish connection of physical entity/card (by calling SRVIXAI)     - Insert data into KCCT.CARDS\_PIN   + Otherwise:     - Failure, update CRDT.PREPAID\_REQ\_ISS accordingly (by calling UCDISS2) * **Otherwise**:   + **For** non rejected records (I\_RECORD.RC=00):     - Update CRDT.PREPAID\_REQ\_ISS with the returned data (by calling UCDISS7)     - In case PAN will remain the same (I\_RECORD.PRODUCTION\_EVENT = 'RRC'), no deletion/connection to SYDIPEL is needed, otherwise:       * Delete connection with SYDIPEL (by calling SRVIXAD)       * Establish connection of legal entity/card (by calling SRVIXAI)       * Establish connection of physical entity/card (by calling SRVIXAI)       * Delete old record from KCCT.CARDS\_PIN       * Insert data into KCCT.CARDS\_PIN   + **Otherwise**:     - Failure, update CRDT.PREPAID\_REQ\_ISS accordingly (by calling UCDISS2) * **If** it is SMARTPIN (I\_RECORD.SMARTPIN\_IND = 'Y'), not a rejected record (I\_RECORD.RC ='00'), and production event PLA (I\_RECORD.PRODUCTION\_EVENT ^= 'PLA'), **then**:   + **If** PAN will be changed (I\_RECORD.PRODUCTION\_EVENT ^= 'RRC'), **then**:     - Insert request in KCCT.SMARTPIN (by calling ICCSPN)     - Insert SMARTPIN track in CRDT.SMARTPIN\_TRACK (by calling ICDSPT)   + Update CRDT.PREPAID\_REQ\_ISS (by calling UCDISS2) * Commit action |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | KCCT.CARDS\_PIN,  CRDT.PREPAID\_REQ\_ISS,  KCCT.SMARTPIN,  CRDT.SMARTPIN\_TRACK |

#### [K77CP] Deblock and Debit/Credit account

(see section [Massive (Un)Loading Voucher](#_d44915nyem41))

#### [K78CP] Perform validations and deliver an output file to IL

(see section [Massive (Un)Loading Voucher](#_7g3fopoj3uog))

#### [K86CP] Archive all data from CRDT.PREPAID\_REQ\_ISS to historical

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | Insert into CRDT.PREPAID\_REQ\_ISS\_HIST all rows from CRDT.PREPAID\_REQ\_ISS for specified type. Delete said rows from CRDT.PREPAID\_REQ\_ISS. |
| **PROGRAM MAIN LOOP** | N/A |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.PREPAID\_REQ\_ISS  CRDT.PREPAID\_REQ\_ISS\_HIST |

## [Disconnect Account from Debit]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Massive Loading Voucher |
| **Batch Flow Title** | KRPPD013-015 & KRPPD017-019 |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | Way4 |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   This process check if accounts of customers with court settlement have to be disconnected from debit card using card and client services. |

### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KRDPD025] Disconnect Account from Debit

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRDPD025** | K60CP | CRDP.PD.CRTSE.FTIP.CRD02502.SEQ | N/A | Validate input file |
|  | K01CP | DFRP.PD.DLDIK.PERP.CLIENTDL.SEQ | CRDP.PD.CRTSE.SEMT.CRD02501.SEQ | Enrich file with SYDIPEL and card PAN |
|  | SORT | CRDP.PD.CRTSE.SEMT.CRD02501.SEQ,  CRDP.PD.CRTSE.FTIP.CRD02502.SEQ | CRDP.PD.CRTSE.SEMT.CRD02503.SEQ | Join file including SYDIPEL and card PAN with Account and Account Details, where PAN is key |
|  | K02CP | CRDP.PD.CRTSE.SEMT.CRD02503.SEQ | CRDP.PD.CRTSE.SEMT.CRD02504.SEQ | Check is accounts of customer with court settlement have to be disconnected from debit card |
|  | SORT | CRDP.PD.CRTSE.SEMT.CRD02504.SEQ | CRDP.PD.CRTSE.FTOP.CRD02505.SEQ | Insert header and trailer to input file |
|  | IEFBR14 | CRDP.PD.CRTSE.SEMT.CRD02501.SEQ,  CRDP.PD.CRTSE.SEMT.CRD02503.SEQ,  CRDP.PD.CRTSE.SEMT.CRD02504.SEQ | N/A | Deallocate working files |

#### 

#### [K60CP] Validate file

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | INPFILE1:  DCL 1 RECFIELD,  5 ROWCODE CHAR (002) INIT(' '),  5 REST\_DETAILS CHAR (67) INIT(' ');    DCL 1 FH\_INFL1 BASED (ADDR(RECFIELD)),  2 FH\_ROWCODE CHAR(02) INIT(' '),  2 FH\_TIMESTP CHAR(26) INIT(' '),  2 FILLER1 CHAR(41) INIT(' ');    DCL 1 FTRL\_INPFILE1 BASED(ADDR(RECFIELD)),  5 FTRL\_ROWCODE CHAR (002) INIT(' '),  5 FTRL\_TOTAL\_RECS PIC'(07)9' INIT(0),  5 FILLER2 CHAR (60) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | INPFILE1\_EOF: Raised when end of file is reached |
| **SERVICES** | SCDDFE: Select from CRDT.DEB\_FILE\_EXC  UCDDFE: Update into CRDT.DEB\_FILE\_EXC |
| **Main Process Logic** | For every line of the input file: Process file records (header, details and trailer.  Perform file validations: If there was no header or trailer, exit If details counter doesn’t align with trailer counter, exit. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * For row code FH (header):   + **If** it’s the first row of the file **then** check if file has already been received for respective date or multiple execution of flow is permitted during single observation date:     - Retrieve latest file execution timestamp and multiple execution indicator (by calling SCDDFE)     - **If** it runs for the first time of the particular date(FH\_INFL1.FH\_TIMESTP>@SCDDFE.ODATA.LATST\_TMSTMP), or multiple execution is permitted for the same date (SUBSTR(L\_FL\_TS,1,10)=SUBSTR(@SCDDFE.ODATA.LATST\_TMSTMP,1,10)) & (@SCDDFE.ODATA.MULT\_EXEC = 'Y')), **then** update latest file execution date (by calling UCDDFE)     - **Else** exit   + **Else** exit * For row code FD (detail): update counter * For row code FT (trailer): update counter, and set respective trailer flag * **Otherwise** Unrecognized, exit |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.DEB\_FILE\_EXC |

#### [K01CP] Enrich file with SYDIPEL and card PAN

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | INPFILE1: File contains SYDIPEL ID marked with court settlement  DCL 1 IN\_REC,  2 RCRA\_I\_IP PIC'(11)9'; |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | OUTFILE1: File contains SYDIPEL ID and respective debit card number  DCL 1 OUT\_REC,  5 O\_CARD CHAR(16),  5 O\_CRA\_I\_IP PIC'(11)9'; |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE: Raised when end of input file is reached  MORE2: Raised when end of output file is reached |
| **SERVICES** | SRVIXAF: fetch from CRAT.C\_IP\_X\_ACC by SYDIPEL |
| **Main Process Logic** | For every line of the input file: Find cards and add data to output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * Read card (by calling SRVIXAF) * For any cards found, iterate through them, and write data to output file |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

#### [K02CP] Check account for court settlement to be disconnected

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | INPFILE1: File contains deposit accounts connected to active debit cards  DCL 1 IN\_REC,  5 RCARD\_NUM CHAR(016),  5 RCRA\_I\_IP PIC'(11)9',  5 RACC\_NUM CHAR(016),  5 RACC\_DESC PIC'(10)9',  5 RACC\_TYP CHAR(002),  5 RACC\_SEQ\_N PIC'(2)9',  5 RACC\_PR\_FL CHAR(001),  5 RACC\_ACTV CHAR(001),  5 RACC\_CON\_TMSMP CHAR(019); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | OUTFILE1: File list of deposit accounts and respective debit card in order to be disconnected in WAY4  DCL 1 O\_REC,  2 O\_CARD CHAR(016),  2 O\_ACCOUNT\_NUM CHAR(016); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE: Raised when end of input file is reached  MORE2: Raised when end of output file is reached |
| **SERVICES** | SCDGLP: Select from CRDT.GL\_ACC\_PARM  SRVAVBA: Retrieve account available balance  SRVSMAS: Retrieve basic information about account  SRVUMAS: Retrieve basic information about account  SELPROD: Select from KATT.PRODUCT |
| **Main Process Logic** | Retrieve operation ID for SRVAVBA and SRVUMAS (by calling SCDGLP).  For every line of the input file: Retrieve ledger balance, pool account system and product code, account type using product code, and check if account should be disconnected before writing data to output file and updating disconnection timestamp. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * Retrieve ledger balance (@SRVAVBA.ODATA.LEDGER\_BALANCE) (by calling SRVAVBA) * Retrieve pool account system (@SRVSMAS.ODATA.POOL\_ACCOUNT\_SYST) and product code (@SRVSMAS.ODATA.PRODUCT\_CODE) (by calling SRVSMAS) * Retrieve account type using product code (@SELPROD.ACCOUNT\_TYPE) (by calling SELPROD) * Check if account should be disconnected (GEN\_ACC = 416 & L\_LEDGER\_BALANCE < 0) or L\_POOL\_ACCOUNT\_SYST = '9'/’8’), and:   + Write data to output file   + Update disconnection timestamp (by calling SRVUMAS) |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.GL\_ACC\_PARM  KATT.PRODUCT |

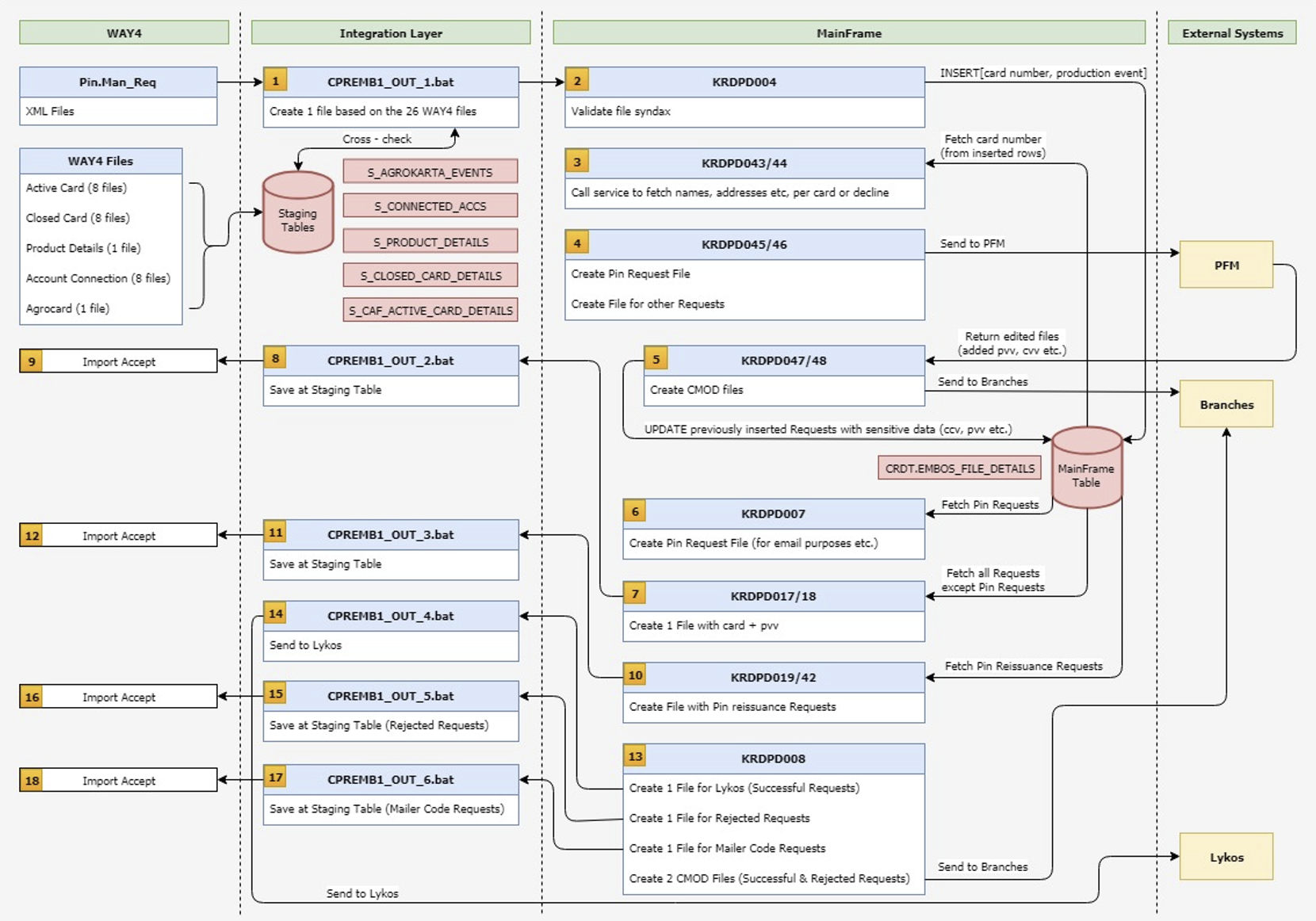
## 

## [Embossing - Debit]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Embossing Debit |
| **Batch Flow Title** | KRDPD004 & KRDPD043 - KRDPD058 & KRDPD017 - KRDPD019 & KRDPD008 |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | Way4 |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   The debit embossing process uses client and cards tables to collect and validate the appropriate data that are required for the process. It sends a file to PFM for the cryptographic card data to be produced. It receives a file from PFM and delivers the embossing file to IL in order to be sent to Inform NEXI or Lykos. |



### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KRDPD004] EMBOSSING PHASE 1 - INSERT DATA INTO RESPECTIVE TABLES

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| KRDPD004 | K07CP | CRDP.PD.DW4PM.FTIP.CRD00401.SEQ | N/A | VALIDATION OF INPUT FILE |
|  | K08CP | CRDP.PD.DW4PM.FTIP.CRD00401.SEQ | DB2 TABLES | INSERTS EMBOSSING DATA INTO DB2 TABLES |
|  | CLEAR01 |  | CRDP.PD.DW4PM.FTIP.CRD00401.SEQ | CLEARS OUT OUTPUT FILE |

#### 

#### [KRDPD043] EMBOSSING PHASE 2 - SUPPLEMENTS TABLES WITH SIDIPEL MASTER DATA (EXCEPT ETHNODEPOSIT)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| KRDPD043 | K43CP | CRDT.EMBOS\_FILE\_DETAIL TABLE | CRDT.EMBOS\_FILE\_DETAIL | SUPPLEMENTS TABLES WITH SIDIPEL MASTER DATA **PARM=’Y’ SMARTPIN** |

#### 

#### [KRDPD044] EMBOSSING PHASE 2 - SUPPLEMENTS TABLES WITH SIDIPEL MASTER DATA (ETHNODEPOSIT)

#### 

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| KRDPD044 | K43CP | CRDT.EMBOS\_FILE\_DETAIL TABLE | CRDT.EMBOS\_FILE\_DETAIL | SUPPLEMENTS TABLES WITH SIDIPEL MASTER DATA **PARM=’N’ SMARTPIN** |

#### 

#### [KRDPD045] EMBOSSING PHASE 3 - CREATE PIN FILE FOR ALL PRODUCTS EXCEPT ETHNODEPOSIT

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| KRDPD045 | K85CP | CRDT.EMBOS\_FILE\_DETAIL | CRDP.PD.DCPFM.FTOP.KCBPIN01.SEQ  CRDP.PD.DCPFM.FTOP.KCBPLA01.SEQ  CRDP.PD.PRVOU.FTOP.PCPFM001.SEQ  CRDP.PD.DCPFM.SEMT.ALLPFM01.SEQ | SELECT KIND IN ('NEW','PLA','PIN','MIG','RWL') WITH COND=’1’  K85CPOU1 (PIN RE-ISSUANCE)  K85CPOU2 ('NEW','PLA','MIG','RWL')  K85CPOU3 ('NEW','PLA','RWL','PIN')  K85CPOU4 ALL PFM RECORDS |
|  | K85CS | CRDP.PD.DCPFM.FTOP.KCBPIN01.SEQ  CRDP.PD.DCPFM.FTOP.KCBPLA01.SEQ | CRDP.PD.DCPFM.PERP.KCBPFM01.SEQ | SORT/MERGE INPUT FILES TO OUTPUT |
|  | K66CP | CRDP.PD.DCPFM.PERP.KCBPFM01.SEQ | CRDP.PD.DCPFM.SEMT.KCBPFM03.SEQ | CREATE OUTPUT WITH ISSUING BRANCH |
|  | K45CP | CRDP.PD.DCPFM.SEMT.ALLPFM01.SEQ | CRDT.EMBO\_FILE\_DETAIL | UPDATE TABLE WITH CONDITION "2" |
|  | K47CS | CRDP.PD.DCPFM.SEMT.KCBPFM03.SEQ | CRDP.PD.DCPFM.SEMT.KCBPFM04.SEQ | SORT BY ISSUING BRANCH |
|  | K47CP | CRDP.PD.DCPFM.SEMT.KCBPFM04.SEQ | CRDP.PD.DCPFM.CMOD.KCBPFM02.SEQ | PRODUCTION OF CMOD REPORT (NEW ISSUANCES /PLASTIC REPLACEMENT) |
|  | CRD44 | CRDP.PD.DCPFM.CMOD.KCBPFM02.SEQ |  | ISSUANCE/RE-ISSUANCE OF DEBIT CARDS |
|  | CLEAR01 |  | CRDP.PD.DCPFM.PERP.KCBPFM01.SEQ  CRDP.PD.DCPFM.PERP.KCCPFM02.SEQ | CLEAR PERP FILES |
|  | DEALLOC | CRDP.PD.DCPFM.SEMT.KCBPFM03.SEQ  CRDP.PD.DCPFM.SEMT.KCBPFM04.SEQ  CRDP.PD.DCPFM.SEMT.ALLPFM01.SEQ |  | DELETE FILES USING IBM UTILITY IEFBR14 - DISP=(OLD,DELETE) |

#### 

#### [KRDPD046] EMBOSSING PHASE 3 - CREATE FILE FOR OTHER REQUESTS

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRDPD046** | K44CP | CRDT.EMBOS\_FILE\_DETAIL | CRDP.PD.DDPFM.FTOP.KDBPIN01.SEQ  CRDP.PD.DDPFM.FTOP.KDBPLA01.SEQ | SELECTS FROM CRDT.EMBOS\_FILE\_DETAIL AND CREATES TWO FILES WITH "PIN" REQUESTS AND INCLUDING THE RESTS |
|  | K45CS | CRDP.PD.DDPFM.FTOP.KDBPIN01.SEQ  CRDP.PD.DDPFM.FTOP.KDBPLA01.SEQ | CRDP.PD.DDPFM.PERP.KDBPFM01.SEQ | SORT/MERGE INPUT TO OUTPUT |
|  | K45CP | CRDP.PD.DDPFM.PERP.KDBPFM01.SEQ | CRDT.EMBOS\_FILE\_DETAIL | UPDATE CRDT.EMBOS\_FILE\_DETAIL TABLE WITH CONDITION=2 |
|  | K46CP | CRDP.PD.DDPFM.PERP.KDBPFM01.SEQ | CRDP.PD.DDPFM.CMOD.KDBPFM02.SEQ | PRODUCTION OF CMOD REPORT |
|  | CRD21 | CRDP.PD.DDPFM.CMOD.KDBPFM02.SEQ |  | REPORTS SUBMISSION |
|  | K48CP | CRDP.PD.DDPFM.PERP.KDBPFM01.SEQ | CRDP.PD.DDPFM.CMOD.KDBPFM03.SEQ | CMOD FILE FOR ETHNODEPOSIT CARDS ONLY |
|  | CRD31 | CRDP.PD.DDPFM.CMOD.KDBPFM03.SEQ |  | BRANCHES DISTRIBUTION REPORT WITH PINS |
|  | CLEAR01 | CRDP.PD.DDPFM.PERP.KDBPFM01.SEQ |  | CLEAR OUT FILE |

#### 

#### [KRDPD047] EMBOSSING PHASE 4 -

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRDPD047** | ALLOC | CRDP.PD.DEBIT.FTOP.CM52POUT.SEQ  CRDP.PD.DEBIT.FTOP.PINML1RE.SEQ |  | ALLOCATE FILES USING IBM UTILITY IEFBR14 – DISP=(NEW,CATLG) |
|  | K49CS0 | CRDP.PD.PRVOU.FTIP.PCPFM001.SEQ | CRDP.PD.PRVOU.PERP.KCCPFM03.SEQ | REFORMAT INPUT TO THE DEBITS FILE |
|  | K49CS | CRDP.PD.DCPFM.FTIP.KCCPIN01.SEQ  CRDP.PD.DCPFM.FTIP.KCCPLA01.SEQ | CRDP.PD.DCPFM.PERP.KCCPFM02.SEQ | SORT/MERGE INPUT TO OUTPUT |
|  | K49CP | CRDP.PD.DCPFM.PERP.KCCPFM02.SEQ  CRDP.PD.PRVOU.PERP.KCCPFM03.SEQ | CRDT.EMBOS\_FILE\_DETAIL | UPDATE CRDT.EMBOS\_FILE\_DETAIL OF CONDITION = 3 TO CONDITION = 2 |
|  | K50CP | CRDP.PD.DCPFM.PERP.KCCPFM02.SEQ | CRDT.EMBOS\_FILE\_DETAIL | UPDATE CRDT.EMBOS\_FILE\_DETAIL OF CONDITION =23 TO CONDITION = 4 |
|  | K51CP | CRDT.EMBOS\_FILE\_DETAIL | CRDT.EMBOS\_FILE\_DETAIL | UPDATE MATURITY DATA ALL WITH CONDITION=’1’ |
|  | K52CP | KCCT.SMARTPIN, CRDT.EMBOS\_FILE\_DETAIL | CRDP.PD.DEBIT.SEMT.CM52POUT.SEQ | UPDATE SMARTPIN/SMARTPIN TRACK |
|  | K52CS | CRDP.PD.DEBIT.SEMT.CM52POUT.SEQ | CRDP.PD.DEBIT.PERP.CM52POUT.SEQ | SORT BY COUNTRY POSTAL CODE |
|  | STEP0001 | CRDP.PD.DEBIT.PERP.CM52POUT.SEQ |  | CHECK EMPTINESS OF INPUT FILE |
|  | PDEMFX | CRDP.PD.DEBIT.PERP.CM52POUT.SEQ | CRDP.PD.DEBIT.CMOD.CM52POUT.SEQ  CRDP.PD.DEBIT.FTOP.PINML1RE.SEQ | CREATION OF PRINTOUT FILE FOR DEBIT CARDS PIN |
|  | BACKSY | CRDP.PD.DEBIT.FTOP.PINML1RE.SEQ | CRDP.PD.DEBIT.GDGT.PINML1RE.BACK(+1) | BACKUP FILE USING IDCAMS REPRO |
|  | COPYFB | CRDP.PD.DEBIT.CMOD.CM52POUT.SEQ | CRDP.PD.DEBIT.FTOP.CM52POUT.SEQ | BACKUP FILE USING IDCAMS REPRO |
|  | BACKF1 | CRDP.PD.DEBIT.FTOP.CM52POUT.SEQ | CRDP.PD.DEBIT.GDGT.CM52POUT.BACK(+1) | BACKUP FILE USING IDCAMS REPRO |
|  | CMPINML | CRDP.PD.DEBIT.CMOD.CM52POUT.SEQ |  | CREATION OF CMOD PRINTOUT |
|  | CLEAR01 |  | CRDP.PD.DCPFM.FTIP.KCCPLA01.SEQ  RDP.PD.PRVOU.FTIP.PCPFM001.SEQ | CLEAR OUT OUTPUT FILES |
|  | DEALLOC | CRDP.PD.DEBIT.SEMT.CM52POUT.SEQ  CRDP.PD.PRVOU.FTOP.PCPFM001.SEQ |  | DELETE INPUT FILES USING IBM UTILITY IEFBR14 |

#### 

#### [KRDPD048] EMBOSSING PHASE 4 FOR ETHNODEPOSIT PRODUCTS ONLY

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRDPD048** | K49CS | CRDP.PD.DDPFM.FTIP.KDCPIN01.SEQ  CRDP.PD.DDPFM.FTIP.KDCPLA01.SEQ | CRDP.PD.DDPFM.PERP.KDCPFM02.SEQ | SORT/MERGE IN ONE FILE PIN AND PLA RESPONSES FROM PFM (ETHNODEPOSIT) |
|  | K49CP | CRDP.PD.DDPFM.PERP.KDCPFM02.SEQ | UPDATE CRDT.EMBOS\_FILE\_DETAIL | BASED ON TRANSACTIONS OF INPUT FILE UPDATE TABLE WITH CONDITION ‘3’ |
|  | K50CP | CRDP.PD.DDPFM.PERP.KDCPFM02.SEQ | CRDT.EMBOS\_FILE\_DETAIL | BASED ON TRANSACTIONS OF INPUT FILE UPDATE TABLE WITH CONDITION ‘4’ |
|  | K51CP | CRDT.EMBOS\_FILE\_DETAIL | CRDT.EMBOS\_FILE\_DETAIL | UPDATE MATURITY TO THE NEXT WORKING DATACRDT.EMBOS\_FILE\_DETAIL COND=’1’ |
|  | CLEAR01 |  | CRDP.PD.DDPFM.PERP.KDCPFM02.SEQ  CRDP.PD.DDPFM.FTIP.KDCPLA01.SEQ | CREAR OUT OUTPUT FILES |

#### 

#### [KRDPD007] UPDATE CORRESPONDING TABLES WITH DETAILS THAT MATRIX HAS BEEN DELIVERED

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRDPD007** | SETPA |  |  | SET PA='CICSAORQ' |
|  | K53CP | KCCT.SMARRTPIN | UPDATE KCCT.SMARTPIN  UPDATE CRDT.SMARTPIN\_TRACK | UPDATE CORRESPONDING TABLES WITH DETAILS THAT MATRIX HAS BEEN DELIVERED |

#### 

#### [KRDPD017] EMBOSSING PHASE 5

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRDPD017** | K09CP | CRDT.EMBOS\_FILE\_DETAIL | CRDP.PD.CPREM.SEMT.CRD01701.SEQ | ALL PRODUCTS EXCEPT ETHNODEPOSIT FLOW (PARM Y) |
|  | K09CS | CRDP.PD.CPREM.SEMT.CRD01701.SEQ | CRDP.PD.CPREM.FTOP.CRD01702.SEQ | SUPPLEMENT OUTPUT WITH HEADER & FOOTER |
|  | DEALLOC | CRDP.PD.CPREM.SEMT.CRD01701.SEQ |  | DELETE FILE USING IBM UTILITY IEFBR14 DISP-(OLD,DELETE) |

#### 

#### 

#### [KRDPD018] EMBOSSING PHASE 5

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRDPD018** | K09CP | CRDT.EMBOS\_FILE\_DETAIL ALL | CRDP.PD.CPREM.SEMT.CRD01801.SEQ | ETHNODEPOSIT FLOW (PARM N) |
|  | K09CS | CRDP.PD.CPREM.SEMT.CRD01801.SEQ | CRDP.PD.CPREM.FTOP.CRD01802.SEQ | SUPPLEMENT OUTPUT WITH HEADER & FOOTER |
|  | DEALLOC | CRDP.PD.CPREM.SEMT.CRD01801.SEQ |  | DELETE FILE USING IBM UTILITY IEFBR14 DISP-(OLD,DELETE) |

#### 

#### 

#### [KRDPD019] EMBOSSING PHASE 6 - ALL PRODUCTS EXCEPT ETHNODEPOSIT FLOW (PARM Y)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRDPD019** | K14CS0 | CRDP.PD.PRVOU.PERP.KCCPFM03.SEQ | CRDP.PD.PRVOU.SEMT.PREPPIN.SE | XTRACT TO OUTPUT PIN REQUESTS |
|  | K14CS1 | CRDP.PD.DCPFM.FTIP.KCCPIN01.SEQ  CRDP.PD.PRVOU.SEMT.PREPPIN.SEQ | CRDP.PD.CPREM.SEMT.CRD01901.SEQ | SORT INPUT FILE AND KEEP ONLY CARD AND ERROR CODE |
|  | K14CP | CRDP.PD.CPREM.SEMT.CRD01901.SEQ | CRDP.PD.CPREM.SEMT.CRD01902.SEQ | SELECTS FROM CRDT.EMBOS\_FILE\_DETAIL AND MASTER AND INSERTS INTO KCCT.TRACK\_INFO |
|  | K14CS2 | CRDP.PD.CPREM.SEMT.CRD01902.SEQ | CRDP.PD.CPREM.FTOP.CRD01903.SEQ | SUPPLEMENT OUTPUT WITH HEADER & FOOTER |
|  | DEALLOC | CRDP.PD.CPREM.SEMT.CRD01901.SEQ  CRDP.PD.CPREM.SEMT.CRD01902.SEQ  CRDP.PD.PRVOU.SEMT.PREPPIN.SEQ |  | DELETE FILE USING IBM UTILITY IEFBR14 DISP-(OLD,DELETE) |
|  | CLEAR01 | CRDP.PD.DCPFM.FTIP.KCCPIN01.SEQ  CRDP.PD.PRVOU.PERP.KCCPFM03.SEQ |  | CLEAR OUT FILE |

#### 

#### 

#### [KRDPD042] EMBOSSING PHASE 6 - ETHNODEPOSIT FLOW (PARM N)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRDPD042** | K14CS1 | CRDP.PD.DDPFM.FTIP.KDCPIN01.SEQ | CRDP.PD.CPREM.SEMT.CRD04201.SEQ | SORT INPUT FILE AND KEEP ONLY CARD AND ERROR CODE |
|  | K14CP | CRDP.PD.CPREM.SEMT.CRD04201.SEQ | CRDP.PD.CPREM.SEMT.CRD04202.SEQ | SELECTS FROM CRDT.EMBOS\_FILE\_DETAIL AND MASTER AND INSERTS INTO KCCT.TRACK\_INFO |
|  | K14CS2 | CRDP.PD.CPREM.SEMT.CRD04202.SEQ | CRDP.PD.CPREM.FTOP.CRD04203.SEQ | SUPPLEMENT OUTPUT WITH HEADER & FOOTER |
|  | DEALLOC | CRDP.PD.CPREM.SEMT.CRD04201.SEQ  CRDP.PD.CPREM.SEMT.CRD04202.SEQ |  | DELETE FILE USING IBM UTILITY IEFBR14 DISP-(OLD,DELETE) |
|  | CLEAR01 | CRDP.PD.DDPFM.FTIP.KDCPIN01.SEQ |  | CLEAR OUT FILE |

#### 

#### [KRDPD008] EMBOSSING PHASE 7 (FOR ALL PRODUCTS)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRDPD008** | ALLOCAT | CRDP.PD.PRVOU.SEMT.CRD04301.SEQ  CRDP.PD.PRVOU.SEMT.CRD04302.SEQ  CRDP.PD.PRVOU.SEMT.CRD04303.SEQ  CRDP.PD.PRVOU.SEMT.CRD04304.SEQ  CRDP.PD.PRVOU.SEMT.CRD04305.SEQ  CRDP.PD.PRVOU.FTOP.CRD04301.SEQ |  | ALLOCATE FILES USING IBM UTILITY IEFBR14 DISP-(NEW,CATLG) |
|  | CLEAR01 | CRDP.PD.PRVOU.PERP.AS400PRE.SEQ |  | CLEAR OUT FILE |
|  | K15CP |  | CRDP.PD.DEBLY.SEMT.CRD00801.SEQ  CRDP.PD.DEBLY.CMOD.CRD00802.SEQ  CRDP.PD.DEBLY.SEMT.CRD00809.SEQ  CRDP.PD.PRVOU.SEMT.CRD00810.SEQ  CRDP.PD.PRVOU.PERP.AS400PRE.SEQ | SELECTS FROM CRDT.EMBOS\_FILE\_DETAIL ALL WITH COND=5 AND WRITES TO EMBOSSING FILE FOR SSIS AND CMOD FILE WITH EXCEPTION RECORDS |
|  | K15CS | CRDP.PD.DEBLY.SEMT.CRD00801.SEQ | CRDP.PD.DEBLY.FTOP.CRD00803.SEQ | SUPPLEMENT OUTPUT WITH HEADER & FOOTER |
|  | K15CS1 | CRDP.PD.DEBLY.SEMT.CRD00808.SEQ | CRDP.PD.DEBLY.FTOP.CRD00810.SEQ | SUPPLEMENT OUTPUT WITH HEADER & FOOTER FOR DEBIT VISA |
|  | K15CS2 | CRDP.PD.PRVOU.SEMT.CRD00810.SEQ | CRDP.PD.PRVOU.FTOP.CRD00811.SEQ | SUPPLEMENT OUTPUT WITH HEADER & FOOTER |
|  | K72CU | CRDP.EMBOS\_FILE\_DETAIL | CRDP.PD.PRVOU.SEMT.CRD04301.SEQ | UNLOAD FROM CRDP.EMBOS\_FILE\_DETAIL CARDS |
|  | K72CS | CRDP.PD.PRVOU.SEMT.CRD04301.SEQ | CRDP.PD.PRVOU.SEMT.CRD04302.SEQ | SORT CARD, IBAN, SYDIPEL USING ICETOOL |
|  | K72CP | CRDP.PD.PRVOU.SEMT.CRD04302.SEQ | CRDP.PD.PRVOU.SEMT.CRD04303.SEQ | VALIDATE ACTIVE ACCOUNTS FROM DISTINCT SYDIPEL |
|  | K73CS | CRDP.PD.PRVOU.SEMT.CRD04301.SEQ  CRDP.PD.PRVOU.SEMT.CRD04303.SEQ | CRDP.PD.PRVOU.SEMT.CRD04304.SEQ | SORT / MERGE |
|  | SORTFIR | CRDP.PD.PRVOU.SEMT.CRD04304.SEQ | CRDP.PD.PRVOU.SEMT.CRD04305.SEQ | SORT ON CARD USING ICETOOL |
|  | K72CP | CRDP.PD.PRVOU.SEMT.CRD04302.SEQ | CRDP.PD.PRVOU.SEMT.CRD04303.SEQ | VALIDATE ACTIVE ACCOUNTS FROM DISTINCT SYDIPEL |
|  | K73CS | CRDP.PD.PRVOU.SEMT.CRD04301.SEQ  CRDP.PD.PRVOU.SEMT.CRD04303.SEQ | CRDP.PD.PRVOU.SEMT.CRD04304.SEQ | SORT USING JOINSKEYS ON CARD AND REFORMAT OUTPUT |
|  | SORTFIR | CRDP.PD.PRVOU.SEMT.CRD04304.SEQ | CRDP.PD.PRVOU.SEMT.CRD04305.SEQ | SELECT UNIQUE RECORDS |
|  | K73CP | CRDP.PD.PRVOU.SEMT.CRD04305.SEQ | CRDP.PD.PRVOU.FTOP.CRD04301.SEQ | EXTRACTS ONLY RECORDS FOR RENEWAL |
|  | K73CU | CRDP.PD.PRVOU.FTOP.CRD04301.SEQ | CRDP.PD.PRVOU.FTOP.CRD04301.SEQ | SUPPLEMENT OUTPUT WITH HEADER & FOOTER |
|  | K16CP | CRDP.PD.DEBLY.SEMT.CRD00809.SEQ | KCCT.TRACK\_INFO | INSERT TO KCCT.TRACK\_INFO ALL RECORDS THAT WILL BE SENT TO LYKOS |
|  | K54CP | CRDP.PD.DEBLY.SEMT.CRD00809.SEQ | CRDT.EMBOS\_FILE\_DETAIL | UPDATE FIELD CONDITION OF CRDT.EMBOS\_FILE\_DETAIL TO ‘6’ |
|  | K55CP | CRDT.EMBOS\_FILE\_DETAIL | CRDP.PD.DEBLY.CMOD.CRD00804.SEQ | CREATE CMOD FILE WITH KIND IN('NEW','PLA','PIN','MIG','RWL') |
|  | K90CP | CRDT.EMBOS\_FILE\_DETAIL | CRDP.PD.DEBLY.CMOD.CRD00812.SEQ | CREATE CMOD FILE FOR VOUCHER |
|  | K41CP | CRDT.EMBOS\_FILE\_DETAIL | CRDP.PD.DEBLY.CMOD.CRD00805.SEQ | CREATE CMOD FOR CARDS SENT TO LYKOS |
|  | K34CP | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER | CRDP.PD.DEBLY.SEMT.CRD00802.SEQ  CRDP.PD.KANON.FTOP.CRD00812.SEQ | COLLECT REJECTED CARDS DURING EMBOSSING OUTGOING PROCESS TO W4 |
|  | K34CS | CRDP.PD.DEBLY.SEMT.CRD00802.SEQ | CRDP.PD.DEBLY.FTOP.CRD00806.SEQ | SUPPLEMENT OUTPUT WITH HEADER & FOOTER |
|  | K37CP | CRDP.PD.DEBLY.SEMT.CRD00809.SEQ | CRDP.PD.DEBLY.SEMT.CRD00803.SEQ | CREATE OUTPUT WITH UNIQUE  MAILER CODE FOR W4 UPDATE |
|  | K37CS | CRDP.PD.DEBLY.SEMT.CRD00803.SEQ | CRDP.PD.DEBLY.FTOP.CRD00807.SEQ | SUPPLEMENT OUTPUT WITH HEADER & FOOTER |
|  | DEALLOC | CRDP.PD.DEBLY.SEMT.CRD00801.SEQ  CRDP.PD.PRVOU.SEMT.CRD04301.SEQ  CRDP.PD.PRVOU.SEMT.CRD04302.SEQ  CRDP.PD.PRVOU.SEMT.CRD04303.SEQ  CRDP.PD.PRVOU.SEMT.CRD04304.SEQ  CRDP.PD.PRVOU.SEMT.CRD04305.SEQ  CRDP.PD.DEBLY.SEMT.CRD00802.SEQ  CRDP.PD.DEBLY.SEMT.CRD00803.SEQ  CRDP.PD.DEBLY.SEMT.CRD00809.SEQ  CRDP.PD.DEBLY.SEMT.CRD00808.SEQ  CRDP.PD.PRVOU.SEMT.CRD00810.SEQ |  | DELETE FILE USING IBM UTILITY IEFBR14 DISP-(OLD,DELETE) |
|  | CRD52 | CRDP.PD.DEBLY.CMOD.CRD00802.SEQ |  | SEND CMOD FILE USING IRXJCL |
|  | CRD32 | CRDP.PD.DEBLY.CMOD.CRD00804.SEQ |  | SEND CMOD FILE USING IRXJCL |
|  | CRD53 | CRDP.PD.DEBLY.CMOD.CRD00805.SEQ CRD053DPL |  | SEND CMOD FILE USING IRXJCL |
|  | CRD60 | CRDP.PD.DEBLY.CMOD.CRD00812.SEQ |  | SEND CMOD FILE USING IRXJCL |

#### [K07CP] - VALIDATION OF INPUT FILE

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | INPFILE1  DCL 1 GD\_INPFILE1,  2 ROWCODE CHAR(02) INIT(' '),  2 BATCH\_ID CHAR(32) INIT(' '),  2 DC\_CARD CHAR(19) INIT(' '),  2 PD\_EVENT CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '),  2 PL\_DELIVERY\_TYPE CHAR(02) INIT(' '),  2 PIN\_DELIVERY\_BRANCH CHAR(05) INIT(' '),  2 ISSUING\_BRANCH CHAR(05) INIT(' '),  2 PE\_EMBOSSING\_NAME CHAR(26) INIT(' '),  2 EMBOSSING\_NAME CHAR(26) INIT(' '),  2 EXPIRATION\_DATE CHAR(10) INIT(' '),  2 TOTAL\_CASH\_WTH\_LMT PIC'(13)9V99' INIT(0),  2 POS\_PURCHASE\_LMT PIC'(13)9V99' INIT(0),  2 TOTAL\_ECOMM\_LMT PIC'(13)9V99' INIT(0),  2 DOM\_WTH\_LMT PIC'(13)9V99' INIT(0),  2 INT\_WTH\_LMT PIC'(13)9V99' INIT(0),  2 DOM\_ECOMM\_LMT PIC'(13)9V99' INIT(0),  2 INT\_ECOMM\_LMT PIC'(13)9V99' INIT(0),  2 ISSUING\_DATE CHAR(10) INIT(' '),  2 CARD\_STATUS CHAR(01) INIT(' '),  2 PIN\_PRODUCT\_STATUS CHAR(01) INIT(' '),  2 PLAST\_PRODUCT\_STATUS CHAR(06) INIT(' '),  2 PLASTIC\_STATUS CHAR(08) INIT(' '),  2 PRODUCT\_CODE CHAR(03) INIT(' '),  2 INSTITUTION\_ID CHAR(04) INIT(' '),  2 SMARTPIN\_ACTIVE CHAR(01) INIT(' '),  2 PRODUCT\_GROUP CHAR(02) INIT(' '),  2 PRODUCT\_NAME CHAR(30) INIT(' '),  2 SCHEME CHAR(10) INIT(' '),  2 BARCODE CHAR(20) INIT(' '),  2 EMPLOYEE\_ID CHAR(20) INIT(' '),  2 DEPARTMENT\_ID CHAR(20) INIT(' '),  2 IBAN CHAR(27) INIT(' '),  /\* B.X. U039 @ 20.10.2019 - START \*/  /\* ΠΡΟΣΘΗΚΗ VISA PREPAID ETE \*/  2 CARD\_TYPE CHAR(01) INIT(' '),  2 COLOR CHAR(01) INIT(' '),  /\* V. PAPPAS - U714 @ 13.12.2022 - START NEW CONSUMER \*/  2 COLOR\_PLASTIC CHAR(01) INIT(' '),  /\* V. PAPPAS - U714 @ 13.12.2022 - END NEW CONSUMER \*/  /\* V. PAPPAS - U714 @ 13.12.2022 - START RECARDING \*/  2 SYDIPEL\_MIG CHAR(11) INIT(' '),  /\* V. PAPPAS - U714 @ 13.12.2022 - END RECARDING \*/  /\* V. PAPPAS - U714 @ 04.05.2023 - START DUAL \*/  2 COMBO\_CARD CHAR(19) INIT(' '),  /\* V. PAPPAS - U714 @ 04.05.2023 - END DUAL \*/  2 FILLER\_DT CHAR(17) INIT(' ');  /\* TO ΠΕΔΙΟ FILLER\_DT ΑΠΟ CHAR(50) ΑΛΛΑΞΕ \*/  /\* ΣΕ CHAR(47) ΓΙΑ ΝΑ ΜΗΝ ΜΕΤΑΒΛΗΘΕΙ ΤΟ \*/  /\* RECSIZE TOY INPUT FILE ΣΤΑ K07CP/K08CP \*/  /\* B.X. U039 @ 20.10.2019 - END \*/  /\* D. MEGREMIS @ 22.02.2018 - END \*/  DCL 1 FH\_INFL1 BASED (ADDR(RECFIELD)),  2 FH\_ROWCODE CHAR(02) INIT(' '),  2 FH\_TIMESTP CHAR(26) INIT(' '),  /\* D. MEGREMIS @ 22.02.2018 - START \*/  /\*2 FILLER1 CHAR(297) INIT(' ');\*/  2 FILLER1 CHAR(424) INIT(' ');  /\* D. MEGREMIS @ 22.02.2018 - END \*/  DCL 1 FTRL\_INPFILE1 BASED (ADDR(RECFIELD)),  2 FTRL\_ROWCODE CHAR(02) INIT(' '),  2 FTRL\_GHEAD\_NO PIC'(07)9' INIT(0),  2 FTRL\_DETAIL\_REC\_NO PIC'(07)9' INIT(0),  /\* D. MEGREMIS @ 22.02.2018 - START \*/  /\*2 FILLER2 CHAR(309) INIT(' ');\*/  2 FILLER2 CHAR(436) INIT(' ');  /\* D. MEGREMIS @ 22.02.2018 - END \*/  DCL 1 GH\_INPFILE1,  2 GH\_ROWCODE CHAR(02) INIT(' '),  2 GH\_FILE\_DATE CHAR(10) INIT(' '),  2 GH\_FILE\_TIME CHAR(08) INIT(' '),  2 GH\_TRANSACTION\_ID CHAR(255) INIT(' '),  2 GH\_FILE\_NUMBER CHAR(06) INIT(' '),  2 GH\_PRODUCTN\_CATEG CHAR(01) INIT(' '),  2 GH\_PRODUCT\_CODE CHAR(03) INIT(' '),  /\* D. MEGREMIS @ 22.02.2018 - START \*/  /\*2 GH\_WAY4\_FILE\_NAME CHAR(40) INIT(' ');\*/  2 GH\_WAY4\_FILE\_NAME CHAR(40) INIT(' '),  2 GH\_FILLER CHAR(127) INIT(' ');  /\* D. MEGREMIS @ 22.02.2018 - END \*/  DCL 1 GTRL\_INPFILE1 BASED (ADDR(RECFIELD)),  2 GTRL\_ROWCODE CHAR(02) INIT(' '),  2 GTRL\_DETAIL\_REC\_NO PIC'(07)9' INIT(' '),  /\* D. MEGREMIS @ 22.02.2018 - START \*/  /\*2 FILLER4 CHAR(316) INIT(' '); \*/  2 FILLER4 CHAR(443) INIT(' ');  /\* D. MEGREMIS @ 22.02.2018 - END \*/  DCL 1 RECFIELD,  2 ROWCODE CHAR(02) INIT(' '),  /\* D. MEGREMIS @ 22.02.2018 - START \*/  /\*2 REST\_DETAILS CHAR(323) INIT(' ');\*/  2 REST\_DETAILS CHAR(450) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | INPFILE1\_EOF :  FOUND\_HEADER  FOUND\_TRAILER |
| **SERVICES** | @SCDDFE : SELECT FROM CRDT.DEB\_FILE\_EXC  @UCDDFE: UPDATE CRDT.DEB\_FILE\_EXC |
| **Main Process Logic** | READS INPUT FILE AND IN A LOOP VALIDATES EXISTENCE/MATCHING OF HEADER/FOOTER DETAIL RE |
| **PROGRAM MAIN LOOP** | READS INPUT FILE AND IN A LOOP VALIDATES IF:   * WHEN ON HEADER CHECK IF IT CAN BE RERUN ON THE SAME DATE OR NOT * WHEN GROUP HEADER (‘GH’) JUST COUNT RECS IN GROUP * WHEN ‘GT’ – GROUP TRAILER CHECK IF RECS OF GROUP ARE THE SAME IN GROUP TRAILER * WHEN ‘FT’ AT END COMPARE DETAIL RECORDS NUMBER AS TO THOSE SPECIFIED IN TRAILER   AT END MAKE ALL CHECKS AS TO THE NUMBER INSIDE AND IN TRAILER  CLOSE FILE AND PRINT STATISTICS |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

#### [K08CP] - UPDATES CRDT.EMBOS\_FILE\_MASTER TABLE WITH DATA RETRIEVED FROM EMBOSSING FILE FOR PIN MANAGEMENT EXPORT FILE

#### 

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | **INPFILE1**  DCL 1 RECFIELD,  2 ROWCODE CHAR (02) INIT(' '),  /\* D. MEGREMIS @ 22.02.2018 - START \*/  /\*2 REST\_DETAILS CHAR (323) INIT(' ');\*/  2 REST\_DETAILS CHAR (450) INIT(' ');  DCL 1 FTRL\_K08CPI BASED(ADDR(RECFIELD)),  2 FTRL\_ROWCODE CHAR (02) INIT(' '),  2 FTRL\_GHEAD\_NO PIC '(07)9' INIT(0),  2 FTRL\_DETAIL\_REC\_NO PIC '(07)9' INIT(0),  /\* D. MEGREMIS @ 22.02.2018 - START \*/  2 FILLER2 CHAR (436) INIT(' ');  /\* D. MEGREMIS @ 22.02.2018 - END \*/  DCL 1 GH\_K08CPI BASED(ADDR(RECFIELD)),  2 GH\_ROWCODE CHAR(02) INIT(' '),  2 GH\_FILE\_DATE CHAR(10) INIT(' '),  2 GH\_FILE\_TIME CHAR(08) INIT(' '),  2 GH\_TRANSACTION\_ID CHAR(255) INIT(' '),  2 GH\_FILE\_NUMBER CHAR(06) INIT(' '),  2 GH\_PRODUCTN\_CATEG CHAR(01) INIT(' '),  2 GH\_PRODUCT\_CODE CHAR(03) INIT(' '),  /\* D. MEGREMIS @ 22.02.2018 - START \*/  /\*2 GH\_WAY4\_FILE\_NAME CHAR(40) INIT(' ');\*/  2 GH\_WAY4\_FILE\_NAME CHAR(40) INIT(' '),  2 GH\_FILLER CHAR(127) INIT(' ');  /\* D. MEGREMIS @ 22.02.2018 - END \*/  DCL 1 GTRL\_K08CPI BASED(ADDR(RECFIELD)),  2 GTRL\_ROWCODE CHAR (02) INIT(' '),  2 GTRL\_DETAIL\_REC\_NO PIC '(07)9' INIT(' '),  /\* D. MEGREMIS @ 22.02.2018 - START \*/  /\*2 FILLER4 CHAR (316) INIT(' '); \*/  2 FILLER4 CHAR (443) INIT(' ');  /\* D. MEGREMIS @ 22.02.2018 - END \*/  DCL 1 GD\_K08CPI BASED(ADDR(RECFIELD)),  2 ROWCODE CHAR(02) INIT(' '),  2 BATCH\_ID CHAR(32) INIT(' '),  2 DC\_CARD CHAR(19) INIT(' '),  2 PD\_EVENT CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '),  2 PL\_DELIVERY\_TYPE CHAR(02) INIT(' '),  2 PIN\_DELIVERY\_BRANCH CHAR(05) INIT(' '),  2 ISSUING\_BRANCH CHAR(05) INIT(' '),  2 PE\_EMBOSSING\_NAME CHAR(26) INIT(' '),  2 EMBOSSING\_NAME CHAR(26) INIT(' '),  2 EXPIRATION\_DATE CHAR(10) INIT(' '),  2 TOTAL\_CASH\_WTH\_LMT PIC '(15)9' INIT(0),  2 POS\_PURCHASE\_LMT PIC '(15)9' INIT(0),  2 TOTAL\_ECOMM\_LMT PIC '(15)9' INIT(0),  /\* A. REPOUSI @ 29.08.2017 - START \*/  2 DOM\_CASH\_WTH\_LMT PIC '(15)9' INIT(0),  2 INT\_CASH\_WTH\_LMT PIC '(15)9' INIT(0),  2 DOM\_ECOMM\_LMT PIC '(15)9' INIT(0),  2 INT\_ECOMM\_LMT PIC '(15)9' INIT(0),  /\* A. REPOUSI @ 29.08.2017 - END \*/  2 ISSUING\_DATE CHAR(10) INIT(' '),  2 CARD\_STATUS CHAR(01) INIT(' '),  2 PIN\_PRODUCT\_STATUS CHAR(01) INIT(' '),  2 PLAST\_PRODUCT\_STATUS CHAR(06) INIT(' '),  2 PLASTIC\_STATUS CHAR(08) INIT(' '),  2 PRODUCT\_CODE CHAR(03) INIT(' '),  2 INSTITUTION\_ID CHAR(04) INIT(' '),  2 SMARTPIN\_ACTIVE CHAR(01) INIT(' '),  2 PRODUCT\_GROUP CHAR(02) INIT(' '),  2 PRODUCT\_NAME CHAR(30) INIT(' '),  /\* D. MEGREMIS @ 22.02.2018 - START \*/  /\*2 FILLER\_DT CHAR(20) INIT(' ');\*/  2 SCHEME CHAR(10) INIT(' '),  2 BARCODE CHAR(20) INIT(' '),  2 EMPLOYEE\_ID CHAR(20) INIT(' '),  2 DEPARTMENT\_ID CHAR(20) INIT(' '),  2 IBAN CHAR(27) INIT(' '),  /\* B.X. U039 @ 20.10.2019 - START \*/  /\* ΠΡΟΣΘΗΚΗ VISA PREPAID ETE \*/  2 CARD\_TYPE CHAR(01) INIT(' '),  2 COLOR CHAR(01) INIT(' '),  /\* V. PAPPAS - U714 @ 13.12.2022 - START NEW CONSUMER \*/  2 COLOR\_PLASTIC CHAR(01) INIT(' '),  /\* V. PAPPAS - U714 @ 13.12.2022 - END NEW CONSUMER \*/  /\* V. PAPPAS - U714 @ 13.12.2022 - START RECARDING \*/  2 SYDIPEL\_MIG CHAR(11) INIT(' '),  /\* V. PAPPAS - U714 @ 13.12.2022 - END RECARDING \*/  /\* V. PAPPAS - U714 @ 04.05.2023 - START DUAL \*/  2 COMBO\_CARD CHAR(19) INIT(' '),  /\* V. PAPPAS - U714 @ 04.05.2023 - END DUAL \*/  2 FILLER\_DT CHAR(17) INIT(' ');  /\* TO ΠΕΔΙΟ FILLER\_DT ΑΠΟ CHAR(50) ΑΛΛΑΞΕ \*/  /\* ΣΕ CHAR(47) ΓΙΑ ΝΑ ΜΗΝ ΜΕΤΑΒΛΗΘΕΙ ΤΟ \*/  /\* RECSIZE TOY INPUT FILE ΣΤΑ K07CP/K08CP \*/  /\* B.X. U039 @ 20.10.2019 - END \*/  /\* D. MEGREMIS @ 22.02.2018 - END \*/ |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | @ICDEFM:INSERT INTO CRDT.EMBOS\_FILE\_MASTER  @SCDEFM1: RETRIEVE MAX FILE\_ID FROM CRDT.EMBOS\_MASTER  @ICDEFD3:INSERT INTO CRDT.EMBOS\_FILE\_DETAIL TABLE  @ICDTRC:INSERT\_ROW\_TRACK\_INFO |
| **Main Process Logic** | UPDATES CRDT.EMBOS\_FILE\_MASTER TABLE WITH DATA RETRIEVED FROM EMBOSSING FILE FOR PIN MANAGEMENT EXPORT FILE |
| **PROGRAM MAIN LOOP** | MAIN PROCESS STEPS ARE:   * OPEN INPUT AND DO THE FIRST READ * IN A LOOP TILL THE EOF OF INPUT   + WHEN ROWCODE= ‘GH’ PREPARE ROW TO CRDT.EMBOS\_FILE\_MASTER . MORE SPECIFICALLY:     - ASSIGN INPUT VALUES TO STRUCTURE: ICDEFM     - BASED ON GH\_PRODUCT\_CODE WHEN:   WHEN ('906') ICDEFM.PRODUCT\_CODE = '106'; (Debit MC Consumer)  WHEN ('916') ICDEFM.PRODUCT\_CODE = '116'; (Debit MC Vanilla)  WHEN ('901') ICDEFM.PRODUCT\_CODE = '201'; (Debit Visa Consumer)  **NOTE**:THE STRUCTURE WILL BE INSERTED WHEN ROWCODE=’GD’ IN NEXT STEP   * + WHEN ROWCODE= ‘GD’     - INSERT ROW TO **CRDT.EMBOS\_FILE\_MASTER** MORE SPECIFICALLY:       * IF EVENT = ‘INS’ (Instant Issuing) AND PRODUCTION\_CODE = ‘A’ (New plastic with new PIN, is used for Mailers to NEXI/Lykos) CHANGE IT TO ‘I’       * IF EVENT = ‘MAS’(Massive Issuance) AND PRODUCT\_CODE IN (106,116,111,112,113) (debit MC, Debit Vanilla, debit mc business self employed, private, premium) AND PRODUCTION CODE = ‘A’ CHANGE IT TO ‘M’.INVOKE SERVICE ICDEFM TO PERFORM THE INSERT     - INSERT ROW TO **CRDT.EMBOS\_FILE\_DETAIL**. MORE SPECIFICALLY:       * ASSIGN VERY MANY INPUT VALUES TO STRUCTURE ICDEFD3       * REGARDING PRODUCT\_CODE:   WHEN ('906') @ICDEFD3.PRODUCT\_CODE = '106';  WHEN ('916') @ICDEFD3.PRODUCT\_CODE = '116';  WHEN ('901') @ICDEFD3.PRODUCT\_CODE = '201';  OTHERWISE @ICDEFD3.PRODUCT\_CODE = GD\_K08CPI.PRODUCT\_CODE (other than 106, 116, 201)   * + - * INSERT USING SERVICE:ICDEFD3     - INSERT ROW TO **KCCT.TRACK\_INFO**. MORE SPECIFICALLY:       * ASSIGN INPUT VALUES TO STRUCTURE: ICDTRC       * WHEN EVENT=’INS’ BASED ON PRODUCT\_GROUP         + **WHEN('01','06','08','09') (debit consumer, credit consumer, prepaid consumer, prepaid KA)** ASSIGN IDATA.HOLDER = TRANSLATE(GD\_K08CPI.EMBOSSING\_NAME,TR2,TR1)         + WHEN('02','07')(debit business, credit business) AND EMBOSSING\_NAME=SPACES ASSIGN IDATA.HOLDER=SPACES ELSE IDATA.HOLDER = TRANSLATE(GD\_K08CPI.PE\_EMBOSSING\_NAME,TR2,TR1);         + WHEN('03')(ethnodeposit) IDATA.HOLDER = ' '         + WHEN('05')(debit self employed) IDATA.HOLDER = TRANSLATE(GD\_K08CPI.PE\_EMBOSSING\_NAME,TR2,TR1);       * ASSIGN VALUES BASED ON PRODUCTION\_CODE         + WHEN A,I,M THEN IDATA.KIND = 'NEW' (new Plastic with new PIN)         + WHEN ‘B’ THEN IDATA.KIND = 'PLA' (only Plastic without new PIN)         + WHEN ‘D’ THEN IDATA.KIND = 'PIN' (only PIN)         + WHEN ‘C’ THEN IDATA.KIND = 'MIG' (new Plastic - migration from one product to an other)         + WHEN ‘E’ THEN IDATA.KIND = RWL (Renewal)       * INSERT USING SERVICE:ICDTRC   + READ NEXT INPUT RECORD * AT END CLOSE INPUT FILE |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_MASTER  CRDT.EMBOS\_FILE\_DETAIL KCCT.TRACK\_INFO |

#### [K43CP] - EMBOSSING PHASE 2 FOR ALL PRODUCTS EXCEPT ETHNODEPOSIT

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | CURSOR ON CRDT.EMBOS\_FILE\_DETAIL WITH KIND IN('NEW','PLA','PIN','MIG','RWL') AND MATURITY EQ TO CURRENT DATE AND COND = ‘0’AND PARM=’Y’(condition = 0 is the starting Point of embossing) |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | SRVIXAS:FIND SIDIPEL FROM CUSTOMER’S CARD  SRVIIPS:FIND MASTER DATA OF A CUSTOMER  SRVILOC:FIND CLASSIFIED ADDRESS DATA  SRVIADR:FIND ADDRESS DATA  SRVINMF:FIND CLIEN’TS NAMES  COUSEL2:FIND MASTER DATA OF A BRANCH  SRVINDS:FIND INDICATORS OF A CLIENT  SRVIXAI:CREATES CARD/CLIENT RELATION  SRVIXAD:DELETE CARD/CLIENT RELATION  KCCSPN:DELETE FROM TABLE KCCT.SMARTPIN  ICDSPT:INSERT IN TABLE CRDT.SMARTPIN\_TRACK  SCDSPT:SELECT IN TABLE CRDT.SMARTPIN\_TRACK  ICCSPN:INSERT IN TABLE KCCT.SMARTPIN  SRVSIBA:FIND ACCOUNT FROM IBAN  SRVIXAE:FINDS RELATION OF CLIENT/ACCOUNT |
| **Main Process Logic** | OPENS CURSOR ON CRDT.EMBOS\_FILE\_DETAIL WITH CERTAIN CRITERIA AND FOR EVERY CARD CHECKS CARD STATUS UPDATES DETAILS OR REMOVES SMARTPIN REQUEST |
| **PROGRAM MAIN LOOP** | THE MAIN STEPS IN THE PROCESSING LOGIC OF THIS PROGRAM ARE:   * GETS CURRENT DATA/TIME * OPEN CURSOR ON CRDT.EMBOS\_FILE\_DETAIL WITH SELECTION CRITERIA KIND IN('NEW','PLA','PIN','MIG','RWL') AND MATURITY\_DATE (is the system SSIS package Date) AS OF CURRENT DATE AND SMARTPIN = ‘Y’ (new PIN is required) * PERFORM THE FIRST FETCH AND IN A LOOP TILL NO MORE FETCHES (100):   + CHECK CARD STATUS AND IF KIND = ‘NEW’ OR ‘MIG’ AND SOME STATUS VALUES ARE NOT MET UPDATE CRDT.EMBOS\_FILE\_DETAIL SO THAT REQUEST IS NOT VALID ANY MORE AND IS REJECTED. These Validations are:     - If (New or MIG) and card\_status! pin\_status!plastic\_production\_status!Plastic status <> ‘02SI’ (active & pin in batch & sent & inactive, respectively) then rejection     - If PIN and card\_status!Pin\_status <> ‘02’ then rejection   + IF NOT UPDATED IN ABOVE STEP:     - SELECT FOR SMARTPIN USING SCDSPT SERVICE AND IF IT IS (‘RWL’ and ‘MIG’) (-that means recarding Visa to MC) and scheme is ‘MC’ INVOKE SERVICE SRVIXAI TO CONNECT TO SYDIPEL AND THEN INVOKE SERVICE ICDSPT TO INSERT TO SMARTPIN\_TRACK AND INVOKE ICCSPN TO INSERT TO SMARTPIN (Recarding Debit Visa to MC)     - IF EVENT = ‘INS’ THEN:       * SELECT BRANCH I\_IP BY INVOKING SERVICE COUSEL2       * WHEN PRODUCT\_GROUP = '01' THEN TYPE = 'ΦΠ' ELSE 'ΝΠ'   + IF EVENT NOT= ‘INS’ THEN:     - RETRIEVE CUSTOMER’S FULL DATA       * SRVIXAS for Customer Sydipel using main cardholder, i.e. Business entity for business cards, consumer sydipel for consumer products       * If product\_code = 029 (voucher prepaid card) & renewal (we need these details for retrieve specific data for the fee charging of business client) then         + Retrieve business account of ΝΠ , using SRVSIBA         + Retrieve consumer of voucher card sydipel, using SRVIXAS       * SRVIIPS for Customer entity: if ETHNODEPOSIT then check if customer is self employed, else reject the record       * SRVILOC & SRVIADR for physical address       * SRVINMF for Customer names         + IF ΝΠ then classification name = (900004 or 900003)         + Else classification name = (900006 or 900001)       * Check onboarding pending, using SRVINDS I\_CLSF\_INDC\_VAL\_K = 8300143       * If renewal then         + Check blocked, using SRVINDS I\_CLSF\_INDC\_VAL\_K = 8800007 and I\_CLSF\_INDC\_VAL\_K = 8800022         + Check under investigation, using SRVINDS I\_CLSF\_INDC\_VAL\_K = 8800007 and I\_CLSF\_INDC\_VAL\_K = 8800024         + Check prohibition, using SRVINDS I\_CLSF\_INDC\_VAL\_K = 8200031 and I\_CLSF\_INDC\_VAL\_K = 8300001         + Check defunct, using SRVIIPS I\_CLSF\_LGL\_CPCTY\_K = 6600020     - IF NOT FULL DATA RETURNED AND THEREFORE THE REQUEST CAN NOT BE COMPLETED UPDATE CRDT.EMBOS\_FILE\_DETAILS TABLE WITH THE REJECTION REASON OF FIRST REASON FOUND   + IF ALL DATA EXIST AND UPDATE CRDT.EMBOS\_FILE\_DETAILS TABLE ACCORDINGLY WITH CONDITION/STATUS SET TO 0   + IN CASE THAT THE CUSTOMER IS MISSING MASTER DATA AND IS RECARDING VISA TO MC AND :     - THERE IS A CARD/CLIENT RELATION, INVOKE SERVICE SRVIXAD TO DELETE CARD/CUSTOMER RELATION     - DELETE KCCT.SMARTPIN BY INVOKING SERVICE KCCSPN   + FETCH NEXT ROW * WHEN END OF CURSOR (100) CLOSE CURSOR AND RETURN |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### 

#### [K44CP] - SELECT FROM CRDT.EMBOS\_FILE\_DETAIL KIND IN ('NEW','PLA','PIN','MIG','RWL') ON CURRENT MATURITY AND OUTPUTS TWO FILES FOR PFM

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | CURSOR ON CRDT.EMBOS\_FILE\_DETAIL |
| **OUTFILE1 FILE DECLARATION** | **K44CPOU1,K44CPOU2**  DCL 1 RFTODPMS,  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME CHAR(30) INIT(' '), /\* ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΚΑΤΟΧΟΥ \*/  2 EXP\_DATE CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '), /\* SERVICE CODE \*/  2 ACTION\_CODE CHAR(01) INIT(' '),  2 MAILER\_CODE CHAR(01) INIT(' '),  2 IDENTITY CHAR(15) INIT(' '), /\* IDENTITY = CRA \*/  2 PIN\_BRANCH CHAR(05) INIT(' '), /\* BRANCH \*/  2 TYP\_REC CHAR(03) INIT(' '), /\* TYPE FILE\*/  2 E\_PIN CHAR(01) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | OPENS CURSOR ON TABLE CRDT.EMBOS\_FILE\_DETAIL WITH CERTAIN CRITERIA AND CURRENT MATURITY DATE, AND FOR EVERY FETCHED ROW CREATES TWO OUTPUT RECORDS |
| **PROGRAM MAIN LOOP** | THE MAIN PROCESSING STEPS IN PROGRAM ARE:   * OPEN CURSOR ON TABLE CRDT.EMBOS\_FILE\_DETAIL AND FETCH 1ST ROW * IN A LOOP TILL NO MORE FETCHES DO:   + CLEAN UP VARIABLES FROM SPECIAL CHARACTERS BY SPACES   + BASED ON CARD KIND ('NEW','PLA','PIN','MIG','RWL') SET DIFFERENT VALUES AS TO MAILER\_CODE(Y/N) AND PIN(Y/N)     - IF ethnodeposit then:       * When New Issuance then Mailer\_code No, Epin Yes       * When PIN then Mailer\_code No, Epin Yes       * When only Plastic then Mailer\_code No, Epin No       * When Renewal then Mailer\_code No, Epin No       * When Migrated then Mailer\_code No, Epin No     - IF ethnocash then:       * When New Issuance then Mailer\_code Yes, Epin No       * When PIN then Mailer\_code Yes, Epin No       * When only Plastic then Mailer\_code No, Epin No       * When Renewal then Mailer\_code No, Epin No       * When Migrated then Mailer\_code No, Epin Yes   + IF KIND OF CARD IS 'NEW','PLA','MIG','RWL'     - OUTPUT CARD TO FILE K44CPOU2     - ELSE TO FILE K44CPOU1   + FETCH NEXT ROW * AT END CLOSE FILES AND PRINT STATS |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### 

#### [K85CP] - SELECT FROM CRDT.EMBOS\_FILE\_DETAIL KIND IN ('NEW','PLA','PIN','MIG','RWL') ON MATURITY LESS OR SAME AS CURRENT AND PRODUCT ‘DC’ AND OUTPUTS FOUR FILES FOR PFM

#### 

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | **K85CPOU2, K85CPOU1**  DCL 1 RFTODPMS,  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME CHAR(30) INIT(' '), /\* ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΚΑΤΟΧΟΥ \*/  2 EXP\_DATE CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '), /\* SERVICE CODE \*/  2 ACTION\_CODE CHAR(01) INIT(' '),  2 MAILER\_CODE CHAR(01) INIT(' '),  2 IDENTITY CHAR(15) INIT(' '), /\* IDENTITY = CRA \*/  2 PIN\_BRANCH CHAR(05) INIT(' '), /\* BRANCH \*/  2 TYP\_REC CHAR(03) INIT(' '), /\* TYPE FILE\*/  2 E\_PIN CHAR(01) INIT(' ');  **K85CPOU3**  DCL 1 VOUCHER\_OUT, /\* 209 CHARACTERS \*/  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME\_1 CHAR(30) INIT(' '), /\* ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΚΑΤΟΧΟΥ \*/  2 CUSTNAME\_2 CHAR(30) INIT(' '),  2 EXP\_DATE CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '), /\* SERVICE CODE \*/  2 PVV CHAR(04) INIT(' '), /\* PIN VERIFICATION VALUE \*/  2 CVC2 CHAR(03) INIT(' '), /\* SERVICE CODE \*/  2 CVC1 CHAR(03) INIT(' '), /\* SERVICE CODE \*/  2 ADDRESS\_1 CHAR(30) INIT(' '), /\* ADDRESS/ADDRESS NO. \*/  2 ADDRESS\_2 CHAR(30) INIT(' '), /\* POSTAL CODE / CITY \*/  2 ACTION\_CODE CHAR(01) INIT(' '),  2 MAILER\_CODE CHAR(01) INIT(' '),  2 ENCRYPTED\_PIN CHAR(05) INIT(' '),  2 ERRORCODE CHAR(02) INIT('00'),  2 SPACE1 CHAR(01) INIT(' '), /\* 'P' \*/  2 SPACE2 CHAR(15) INIT(' '), /\* 'PREPAID CARD' \*/  2 SPACE3 CHAR(03) INIT(' '), /\* KIND \*/  2 SPACE4 CHAR(01) INIT(' '),  2 SPACE5 CHAR(11) INIT(' '), /\* LEGAL SIDIPEL \*/  2 SPACE6 CHAR(15) INIT(' '), /\* BRANCH \*/  2 E\_PIN CHAR(01) INIT('Y'); /\* ALWAYS 'Y' \*/  /\*\*\*\*\*\* CMS - I.ATTIKOS - 01.05.2018 - END \*\*\*\*\*\*\*\*\*\*/  **K85CPOU4**  DCL 1 ALLPFM,  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME CHAR(30) INIT(' '),  2 EXP\_DATE CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '),  2 ACTION\_CODE CHAR(01) INIT(' '),  2 MAILER\_CODE CHAR(01) INIT(' '),  2 IDENTITY CHAR(15) INIT(' '),  2 PIN\_BRANCH CHAR(05) INIT(' '),  2 TYP\_REC CHAR(03) INIT(' '),  2 E\_PIN CHAR(01) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | FOR EVERY ROW OF THE CURSOR DEPENDING ON THE DETAIL PRODUCT GROUP, AS TO VOUCHER PRODUCT GROUP (‘04’) OR DEBIT PRODUCT GROUP, IS ANALYZED AND WRITTEN IN A SPECIFIC FILE FORMAT TO PFM |
| **PROGRAM MAIN LOOP** | MAIN STEPS THAT ARE PERFORMED ARE:   * OPEN CURSOR AND FETCH 1ST ROW * FOR EVERY FETCHED ROW AND BASED ON DETAIL PRODUCT GROUP :   + IF THE ROW IS REFERRED TO VOUCHER (with product\_group = ‘04’)     - Mailer\_code No, Epin Yes     - WRITE TO VOUCHER FILE FOR PFM   + ELSE DEBIT:     - IF ethnodeposit then:       * When New Issuance then Mailer\_code Yes, Epin No       * When PIN then Mailer\_code Yes, Epin No       * When only Plastic then Mailer\_code No, Epin No       * When Renewal then Mailer\_code No (only when card is recarding then Mailer\_code Yes), Epin No       * When Migrated then Mailer\_code No, Epin No     - else:       * When New Issuance then Mailer\_code No, Epin Yes       * When PIN then Mailer\_code No, Epin Yes       * When only Plastic then Mailer\_code No, Epin No       * When Renewal then Mailer\_code No, Epin No (only when card is recarding then Epin Yes)       * When Migrated then Mailer\_code No, Epin Yes     - WRITE TO DEBIT FILE FOR PFM   + WRITE TO FILE WITH ALL RECORDS (E.G. DEBIT, VOUCHER) (The 3rd file) * FETCH NEXT ROW * AT END CLOSE ALL FILES AND PRINT STATS |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### [K66CP] - RETRIEVE ISSUING BRANCH FOR EACH NEW REQUEST

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | **INPFILE1**  DCL 1 I\_RFTODPMS,  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME CHAR(30) INIT(' '),/\* ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΚΑΤΟΧΟΥ\*/  2 EXP\_DATE CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '), /\* SERVICE CODE \*/  2 ACTION\_CODE CHAR(01) INIT(' '),  2 MAILER\_CODE CHAR(01) INIT(' '),  2 IDENTITY CHAR(15) INIT(' '), /\* IDENTITY = CRA \*/  2 PIN\_BRANCH CHAR(05) INIT(' '), /\* BRANCH \*/  2 TYP\_REC CHAR(03) INIT(' '), /\* TYPE FILE\*/  2 E\_PIN CHAR(01) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | **OUTFILE1**  DCL 1 O\_RFTODPMS,  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME CHAR(30) INIT(' '),/\* ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΚΑΤΟΧΟΥ\*/  2 EXP\_DATE CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '), /\* SERVICE CODE \*/  2 ACTION\_CODE CHAR(01) INIT(' '),  2 MAILER\_CODE CHAR(01) INIT(' '),  2 IDENTITY CHAR(15) INIT(' '), /\* IDENTITY = CRA \*/  2 ISSUING\_BRANCH CHAR(05) INIT(' '), /\* BRANCH \*/  2 TYP\_REC CHAR(03) INIT(' '), /\* TYPE FILE\*/  2 E\_PIN CHAR(01) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | SCDEFD2: RETRIEVE ISSUING BRANCH |
| **Main Process Logic** | FOR EVERY INPUT RECORD RETRIEVE ISSUING BRANCH AND OUTPUT IT |
| **PROGRAM MAIN LOOP** | MAIN STEPS IN PROGRAM ARE:   * OPEN INPUT FILE AND FETCH FIRST ROW * IN LOOP PERFORM THE FOLLOWING STEPS   + INITIALIZE OUTPUT AREA   + CALL SERVICE TO RETRIEVE ISSUING BRANCH, using SCDEFD2   + ASSIGN VALUES TO OUTPUT AREAS   + WRITE RECORD TO FILE OUTFILE1   + READ NEXT RECORD * AT END CLOSE FILES |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

#### [K45CP] - UPDATES CRDT.EMBOS\_FILE\_DETAIL TO CONDITION AND STATUS (2) FOR EVERY INPUT CARD

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | **K45CPIN**  DCL 1 RK45CPIN,  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME CHAR(30) INIT(' '), /\* ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΚΑΤΟΧΟΥ \*/  2 EXP\_DATE CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '), /\* SERVICE CODE \*/  2 ACTION\_CODE CHAR(01) INIT(' '),  2 MAILER\_CODE CHAR(01) INIT(' '),  2 IDENTITY CHAR(15) INIT(' '), /\* IDENTITY \*/  2 PIN\_BRANCH CHAR(05) INIT(' '), /\* BRANCH \*/  2 TYP\_REC CHAR(03) INIT(' '),  2 E\_PIN CHAR(01) INIT(' '); /\* TYPE FILE\*/ |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | SCDTRC: RETRIEVE LATEST TRACK OF CARD  ICDTRC: INSERT RECORDS TO TRACKING TABLE |
| **Main Process Logic** |  |
| **PROGRAM MAIN LOOP** | THE MAIN PROCESSING STEPS ARE:   * OPENS INPUT FILE AND READ FIRST RECORD * IN A LOOP TILL THE END OF FILE PROFORM THE FOLLOWING STEPS   + COUNT 'NEW','PLA','MIG','RWL'   + COUNT 'PIN'   + SELECT CARD DETAILS AS IN TABLE CRDT.EMBOS\_FILE\_DETAIL   + DETERMINE EC->ΕΘΝΟCASH/ED->ΕΘΝΟDEPOSIT   + PERFORM UPDATE OF CRDT.EMBOS\_FILE\_DETAIL BY SETTING CONDITION AND STATUS TO 2 (condition = 2 that the process is in the PFM to generate security data)   + RETRIEVE LATEST TRACK OF CARD FROM THE TRACK INFO TABLE (SERVICE SCDTRC)   + INSERT RECORDS TO TRACKING TABLE BY INVOKING SERVICE ICDTRC   + READ NEXT RECORD * AT END CLOSE FILE |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### [K47CP] - CREATES A REPORT FILE BASED ON THE FILE SENT TO PFM

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | **K47CPIN**  DCL 1 RK47CPIN,  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME CHAR(30) INIT(' '), /\* ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΚΑΤΟΧΟΥ \*/  2 EXP\_DATE CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '), /\* SERVICE CODE \*/  2 ACTION\_CODE CHAR(01) INIT(' '),  2 FILLER CHAR(01) INIT(' '),  2 IDENTITY CHAR(15) INIT(' '), /\* IDENTITY \*/  2 PIN\_BRANCH0 CHAR(02) INIT(' '), /\* BRANCH KENO \*/  2 PIN\_BRANCH CHAR(03) INIT(' '), /\* BRANCH \*/  2 TYP\_REC CHAR(03) INIT(' '), /\* TYPE FILE\*/  2 E\_PIN CHAR(01) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | **K47CPOUT**  DCL FLINE CHAR(133) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | CREATES A CMOD FILE WITH NEW ISSUANCES AND PLASTIC REPLACEMENT CARDS |
| **PROGRAM MAIN LOOP** | THE MAIN STEPS THAT ARE PERFORMED ARE:   * OPEN INPUT AND READ FIRST RECORD * WHEN REC\_TYPE ‘'NEW','PLA' SELECT CARD DETAILS FROM CRDT.EMBOS\_FILE\_DETAIL AND CONDITION =2 * IF THERE IS A BRANCH CHANGE PRINT HEADINGS * IDENTIFY PRODUCTION EVENT AND CHARACTERIZE IT (ΕΠΑΝΕΚΔΟΣΗ/ΝΕΑ ΕΚΔΟΣΗ) * WRITE K47CPOUT * READ NEXT INPUT RECORD * AT END CLOSE FILES |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### 

#### [K46CP] - PHASE E, NEW ISSUANCE/RE-ISSUANCE OF CARD FOR PIN/NEW/MIG

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | **K46CPIN**  DCL 1 RK46CPIN,  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME CHAR(30) INIT(' '), /\* ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΚΑΤΟΧΟΥ \*/  2 EXP\_DATE CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '), /\* SERVICE CODE \*/  2 ACTION\_CODE CHAR(01) INIT(' '),  2 MAILER\_CODE CHAR(01) INIT(' '),  2 IDENTITY CHAR(15) INIT(' '), /\* IDENTITY \*/  2 PIN\_BRANCH CHAR(05) INIT(' '), /\* BRANCH \*/  2 TYP\_REC CHAR(03) INIT(' '), /\* TYPE FILE\*/  2 E\_PIN CHAR(01) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | **K46CPOUT**  DCL FLINE CHAR(133) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | TPROD:PRODUCT CODES  TPRODNAME:PRODUCT NAMES  TPRODCNT:COUNTERS |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** |  |
| **PROGRAM MAIN LOOP** | MAIN PROCESSING STEPS ARE:   * GET CURRENT DATE * OPEN CURSOR AND DISTINCTIVELY RETRIEVE PRODUCTS INTO TPROD AND TPRODNAME ARRAYS * BASED ON ‘EIDOS’ CHARACTERIZE ΕΘΝΟCASH OR ΕΘΝΟDEPOSIT * OPEN INPUT & OUTPUT FILES AND MAKE THE FIRST READ * IN A LOOP PERFORM THE FOLLOWING STEPS:   + WHEN CARD TYPE 'NEW','PIN','MIG'     - RETRIEVE CARD DETAILS FROM CRDT.EMBOS\_FILE\_DETAIL AND CONDITION = 2     - BASED ON CARD CHARACTERISTICS DETERMINE TPRODCNT INDEXES     - DETERMINE BRANCH CHARACTERISTICS     - WRITE CMOD FILE   + WHEN CARD TYPE ‘PLA’     - RETRIEVE CARD DETAILS FROM CRDT.EMBOS\_FILE\_DETAIL AND CONDITION = 2     - BASED ON CARD CHARACTERISTICS DETERMINE TPRODCNT INDEXES   + PERFORM NEXT INPUT FILE READ * AT END CLOSE FILES * PRINT TOTALS |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### [K48CP] - CREATES A CMOD FILE BASED OF PFM FILE FOR NEW ISSUANCES OR REPLACEMENTS

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | **K48CPIN**  DCL 1 RK48CPIN,  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME CHAR(30) INIT(' '), /\* ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΚΑΤΟΧΟΥ \*/  2 EXP\_DATE CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '), /\* SERVICE CODE \*/  2 ACTION\_CODE CHAR(01) INIT(' '),  2 FILLER CHAR(01) INIT(' '),  2 IDENTITY CHAR(15) INIT(' '), /\* IDENTITY \*/  2 PIN\_BRANCH0 CHAR(02) INIT(' '), /\* BRANCH KENO \*/  2 PIN\_BRANCH CHAR(03) INIT(' '), /\* BRANCH \*/  2 TYP\_REC CHAR(03) INIT(' '), /\* TYPE FILE\*/  2 E\_PIN CHAR(01) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | DCL FLINE CHAR(133) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | CREATES AN OUTPUT FILE FOR CMOD BASED ON THE EXPORTED FILE FOR PFM |
| **PROGRAM MAIN LOOP** | THE MAIN PROCESSING STEPS OF THE PROGRAM ARE:   * GET CURRENT DATE * OPEN FILES AND GET THE FIRST RECORD * IN A LOOP TILL END OF INPUT FILE PERFORM THE FOLLOWING TASKS:   + WHEN CARD TYPE IS 'NEW','PIN','MIG'     - GET CARD DETAILS FROM CRDT.EMBOS\_FILE\_DETAIL     - IF THERE IS A BRANCH CHANGE PRINT NEW HEADINGS     - WRITE CMOD FILE (K48CPOUT) WITH ΝΕΑ ΕΚΔΟΣΗ/ΕΠΑΝΕΚΔΟΣΗ PIN/MIGRATION     - READ NEXT RECORD * CLOSE OUTPUT FILES |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### [K49CP] - BASED ON INPUT FILE UPDATE TABLE CRDT.EMBOS\_FILE\_DETAIL (CONDITION AND STATUS)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | **K49CPIN**  DCL RRK49CPIN CHAR(96);  DCL 1 RK49CPIN UNALIGNED BASED(ADDR(RRK49CPIN)),  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME CHAR(30) INIT(' '),  2 PVV CHAR(04) INIT(' '),  2 CVV2 PIC'(03)9' INIT(0),  2 CVV1 PIC'(03)9' INIT(0),  2 ICVV PIC'(03)9' INIT(0),  2 PIN\_BLOCK CHAR(16) INIT(' '),  2 ERROR\_CDE CHAR(02) INIT(' '), /\*ERROR CODE \*/  2 TYP\_REC CHAR(03) INIT(' '), /\*NEW,PIN,PLA,MIG\*/  2 E\_PIN CHAR(01) INIT(' '),  2 MATRIX CHAR(15) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | UPDATES CRDT.EMBOS\_FILE\_DETAIL OF CONDITION =3 TO 2 WHEN KIND IN ('NEW','PLA','PIN','MIG','RWL') |
| **PROGRAM MAIN LOOP** | THE MAIN PROCESSING STEPS OF THE PROGRAM ARE:   * INITIALIZE TO CONDITION = 2 THOSE CARDS THAT HAVE CONDITION 3, due to program re - execution, AND KIND IN 'NEW','PLA','PIN','MIG','RWL' OF TABLE CRDT.EMBOS\_FILE\_DETAIL * OPEN INPUT FILE AND READ FIRST RECORD * RETRIEVE CARD DETAILS FROM CRDT.EMBOS\_FILE\_DETAIL * VALIDATE DETAIL CONDITIONS AND ISSUE ERROR IF NECESSARY * BASED ON RECORD TYPE UPDATE COUNTERS FOR GROUPS OF NEW PIN, PLA, MIG, RWL * UPDATE CARD CONDITION AND STATUS (CRDT.EMBOS\_FILE\_DETAIL) TO 3 * FETCH NEXT ROW * AT END CLOSE FILES AND PRINT TOTALS |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### 

#### [K50CP] - BASED ON FILE EXPORTED FOR PFM UPDATE DETAILS, SMARTPIN AND TRACK TABLES

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | **K50CPIN**  DCL 1 RK50CPIN,  2 CARD CHAR (16) INIT(' '),  2 CUSTNAME CHAR (30) INIT(' '),  2 PVV CHAR (04) INIT(' '),  2 CVV2 PIC '(03)9' INIT(0),  2 CVV1 PIC '(03)9' INIT(0),  2 ICVV PIC '(03)9' INIT(0),  2 PIN\_BLOCK CHAR (16) INIT(' '),  2 ERROR\_CDE CHAR (02) INIT(' '),  2 TYP\_REC CHAR (03) INIT(' '),  2 E\_PIN CHAR (01) INIT(' '),  2 MATRIX CHAR (15) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | BASED ON INPUT FILE OF PFM UPDATE DETAILS, SMARTPIN AND TRACK DATA TABLES |
| **PROGRAM MAIN LOOP** | THE MAIN PROCESSING STEPS OF THE PROGRAM ARE:   * GET CURRENT DATE * CHECK ALL THE INCOMPLETE REQUESTS, BY using a sql query in to EMBOS\_FILE\_DETAIL table with condition 2, which means PFM (Pin File Management) didn’t process the card and its mailer code and pin. * OPEN FILES AND GET THE FIRST RECORD * IN A LOOP TILL END OF INPUT FILE PERFORM THE FOLLOWING TASKS:   + SELECT CARD DETAILS CRDT.EMBOS\_FILE\_DETAIL   + WHEN PARM = ‘Y’ AND TYPE <> ‘PLA’ AND <> ‘RWL’ SELECT FORM CRDT.SMARTPIN\_TRACK DATA   + WHEN PARM = ‘Y’ AND TYPE = ‘RWL’ OR ‘MIG’ SELECT FORM CRDT.SMARTPIN\_TRACK DATA   + WHEN CARD DETAILS CONDITION = 3 THEN   + UPDATE STATUS/CONDITION TO 4, CVV1/CVV2/ICVV PIN BLOCK at EMBOS\_FILE\_DETAIL     - WHEN PARM = ‘Y’ AND TYPE REC = NEW,PIN,MIG,PLA UPDATE SMARTPIN     - WHEN PARM = ‘Y’ AND TYPE REC = PLA DO NOTHING     - WHEN PARM = ‘Y’ AND TYPE REC = RWL UPDATE SMARTPIN   + READ NEXT INPUT RECORD * AT END, CLOSE FILES AND RETURN TO THE OPERATING SYSTEM |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### 

#### [K51CP] - RETRIEVE CRDT.EMBOS\_FILE\_DETAIL FOR CONDITION=1 AND UPDATE DETAILS TABLE WITH MATURITY DAY THE NEXT WORKING DATA

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | SELECT CARD  FROM CRDT.EMBOS\_FILE\_DETAIL  WHERE KIND IN('NEW','PLA','PIN','RWL')  AND MATURITY\_DATE = :WS\_DATE\_CUR  AND CONDITION = 1  AND SMARTPIN = :EIDOS  AND CLASS\_PRODUCT = 'DC' |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | RETRIEVE CRDT.EMBOS\_FILE\_DETAIL FOR CONDITION=1 AND UPDATE DETAILS TABLE WITH MATURITY DAY THE NEXT WORKING DATA |
| **PROGRAM MAIN LOOP** | THE MAIN PROCESSING STEPS OF THE PROGRAM ARE:   * GET CURRENT DATE * EVALUATE PARM   + WHEN Y->ΕΘΝΟCASH   + WHEN N->ΕΘΝΟDEPOSIT * COUNT FROM CRDT.EMBOS\_FILE\_DETAIL AND CONDITION 1 ALL WITH KIND = 'NEW','PLA','PIN','RWL' AND IF EXIST ISSUE WARNING MESSAGES * FIND NEXT WORKING DAY & DATE AFTER TODAY * OPEN CURSOR FOR CRDT.EMBOS\_FILE\_DETAIL AND KIND IN('NEW','PLA','PIN','RWL') AND CONDITION = 1 AND FETCH FIRST ROW * IN A CURSOR LOOP :   + UPDATE CRDT.EMBOS\_FILE\_DETAIL MATURITY DATE AFTER ONE WORKING DAY   + FETCH NEXT CURSOR ROW * AT END CLOSE CURSOR AND RETURN TO OPERATING SYSTEM |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### 

#### [K52CP] - RETRIEVES ALL REQUESTS FROM KCCT.SMARTPIN THAT NEED SMART PIN REISSUANCE (REQUEST = "3") VIA MAIL AND SMARTPIN REMINDER (REQUEST = "1") VIA MAIL AND UPDATE MATRIX\_SEND\_STATUS AND MATRIX\_SEND\_DATE AND MATRIX\_DELIVERY\_TYPE TO "1"

#### 

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | SELECT \*  FROM KCCT.SMARTPIN  WHERE CARD\_ORIGIN = 'DC'  AND REQUEST IN ('1','3')  AND FLG\_DELIVER IN ('0','1')  AND MATRIX ^= ' '  AND FLG = '2' |
| **OUTFILE1 FILE DECLARATION** | **K52CPOUT**  DCL 1 RK52CPOUT,  2 CARD CHAR (16) INIT(' '),  2 FIL01 CHAR (01) INIT(' '),  2 REQUEST CHAR (01) INIT(' '), /\* '1' REISSUE,  '3' NEW \*/  2 FIL02 CHAR (01) INIT(' '),  2 REQ\_DATE CHAR (10) INIT(' '),  2 FIL03 CHAR (01) INIT(' '),  2 SND\_DATE CHAR (10) INIT(' '),  2 FIL04 CHAR (01) INIT(' '),  2 HOLDER CHAR (26) INIT(' '),  2 FIL05 CHAR (01) INIT(' '),  2 CUST\_NAME CHAR (40) INIT(' '),  2 FIL06 CHAR (01) INIT(' '),  2 COMP\_NAME CHAR (40) INIT(' '),  2 FIL07 CHAR (01) INIT(' '),  2 CRA\_ADDR CHAR (40) INIT(' '),  2 FIL08 CHAR (01) INIT(' '),  2 CRA\_TK CHAR (40) INIT(' '),  2 FIL09 CHAR (01) INIT(' '),  2 CRA\_CITY CHAR (40) INIT(' '),  2 FIL10 CHAR (01) INIT(' '),  2 CRA\_COUNTRY CHAR (40) INIT(' '),  2 FIL11 CHAR (01) INIT(' '),  2 MATRIX CHAR (15) INIT(' '),  2 FIL12 CHAR (01) INIT(' '),  2 FLAG\_ENT CHAR (02) INIT('NO'), /\* ΕΝΘΕΤΟ \*/  2 FIL13 CHAR (01) INIT(' '),  2 FLAG\_LET CHAR (02) INIT('NO'), /\* ΕΠΙΣΤΟΛΗ \*/  2 FIL14 CHAR (01) INIT(' '),  2 CARD\_ORIGIN CHAR (02) INIT(' '), /\* CC,DC \*/  2 FIL15 CHAR (01) INIT(' '),  2 PRODUCT CHAR (09) INIT(' '),  2 FIL16 CHAR (01) INIT(' '),  2 PRODUCT\_NAME CHAR (30) INIT(' '),  2 FIL17 CHAR (01) INIT(' '),  2 CNT PIC '(6)9' INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | SRVINMF:  SRVIXAL:  UCDSPT2: |
| **Main Process Logic** | RETRIEVES KCCT.SMARTPIN TABLE ALL REQUESTS THAT NEED SMART PIN REISSUANCE (REQUEST = "3") VIA MAIL AND SMARTPIN REMINDER (REQUEST = "1") VIA MAIL DELIVERY |
| **PROGRAM MAIN LOOP** | THE MAIN PROCESSING STEPS OF THE PROGRAM ARE:   * GET CURRENT DATE/TIME * OPEN OUTPUT FILE K52CPOUT * OPEN CURSOR AND FETCH FIRST * IN LOOP INSIDE CURSOR, PERFORM THE FOLLOWING:   + UPDATE KCCT.SMARTPIN COLUMNS FLG/MATRIX FORMATING MATRIX VALUE BY SERVICE RKCCSPIN   + UPDATE KCCT.SMARTPIN TABLE BY:     - RETRIEVE FIRST FROM CRDT.SMARTPIN\_TRACK     - UPDATE KCCT.SMARTPIN BY SERVICE UCDSPT2   + IF DELIVERY TYPE IS NOT VIA BRANCH, THEN     - COLLECT CUSTOMER'S DATA BY SERVICES SRVIXAL/SRVINMF     - WRITE TO CMOD OUTPUT FILE   + PERFORM NEXT FETCH * WHEN NO MORE FETCHED ROWS CLOSE CURSOR AND FILES AND RETURN |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.SMARTPIN\_TRACK |

#### 

#### [K53CP] - PROCESS ALL REQUESTS FOR SMARTPIN REISSUANCE (MATRIX DELIVERY) VIA EMAIL (USE **EXCI:EXTERNAL CICS INTERFACE TECHNOLOGY**)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | SELECT \*  FROM KCCT.SMARTPIN A,  CRDT.EMBOS\_FILE\_DETAIL B  WHERE A.CARD = B.CARD  AND A.CARD\_ORIGIN IN ('DC','PC')  AND A.REQUEST = '3'  AND A.FLG\_DELIVER = '2'  AND A.FLG = '2'  AND A.MATRIX ^= ' '  AND B.MATURITY\_DATE <= :WS\_DATE\_CUR  AND CLASS\_PRODUCT = 'DC'; |
| **OUTFILE1 FILE DECLARATION** | **N/A** |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | PROCESS ALL REQUESTS FOR SMARTPIN REISSUANCE (MATRIX DELIVERY) VIA EMAIL (USE **EXCI:EXTERNAL CICS INTERFACE TECHNOLOGY**) AND UPDATE TABLES KCCT.SMARTPIN  CRDT.SMARTPIN\_TRACK ACCORDINGLY |
| **PROGRAM MAIN LOOP** | THE MAIN PROCESSING STEPS OF THE PROGRAM ARE:   * GET CURRENT DATE/TIME * OPEN CURSOR AND FETCH FIRST * IN LOOP INSIDE CURSOR, PERFORM THE FOLLOWING:   + INVOKE CROSMS CICS PROGRAM TO SEND EMAIL   + IF EXCI LINK WAS SUCCESSFUL UPDATE KCCT.SMARTPIN TABLE BY SETTING FLG = '1'   + RETRIEVE SMARTTRACK INFO FROM CRDT.SMARTPIN\_TRACK   + UPDATE CRDT.SMARTPIN\_TRACK BY SETTING MATRIX\_SEND\_STATUS = '1' AND MATRIX\_SEND\_DATE TO THE CURRENT DATE   + FETCH NEXT ROW * AT END OF CURSOR ROWS CLOSE CURSOR AND RETURN |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | KCCT.SMARTPIN  CRDT.SMARTPIN\_TRACK |

#### 

#### [K09CP] - SELECTS FROM CRDT.EMBOS\_FILE\_DETAIL WITH KIND **NOT** IN ('PIN') AND CONDITION=4 AND WRITES FILE FOR PVV

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | **N/A** |
| **OUTFILE1 FILE DECLARATION** | **K09CPOU1**  DCL 1 O\_PINRESPONSE,  2 O\_CARD CHAR(19) INIT(' '),  2 O\_PVV CHAR(04) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | SELECTS FROM CRDT.EMBOS\_FILE\_DETAIL WITH KIND **NOT** IN ('PIN') AND CONDITION=4 AND WRITES FILE FOR PMS. UPDATE PROCESSED CARDS TO STATUS/CONDITION = 5 |
| **PROGRAM MAIN LOOP** | THE MAIN PROCESSING STEPS OF THE PROGRAM ARE:   * OPEN OUTPUT FILE * OPEN CURSOR AND FETCH FIRST * IN A LOOP TILL END OF CURSOR PERFORM THE FOLLOWING TASKS:   + IF KIND = ‘NEW’ OR 'MIG' OR (‘RWL’ AND PRODUCTION\_EVENT = 'MIG', WRITE TO THE OUTPUT K09CPOU1, CARD & PVV   + UPDATE TABLE DETAIL WITH CONDITION/STATUS=5 BY USING SERVICE UCDEFD1   + COMMIT EVERY 100 ROWS   + FETCH NEXT * AT END CLOSE CURSOR & OUTPUT FILE * RETURN TO O/S |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### 

#### [K14CP] - BASED ON INPUT FILE SELECTS FROM CRDT.EMBOS\_FILE\_DETAIL PIN REISSUANCES WITH CONDITION=4 AND UPDATE MASTER/DETAIL AND INSERT INTO TRACK\_INFO

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | **K14CPIN1**  DCL 1 RK14CPIN1 ,  2 CARD CHAR (019) INIT(' '),  2 ERROR\_CDE CHAR (002) INIT(' '),  2 KND CHAR (003) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | **K14CPOU1**  DCL 1 PINRESPONSE, /\* 352 \*/  2 O\_WAY4FILENAME CHAR (040) INIT(' '),  /\* 2 O\_TRANSACTION\_ID CHAR (010) INIT(' '), \*/  2 O\_TRANSACTION\_ID CHAR (255) INIT(' '),  /\* 2 O\_BATCH\_ID CHAR (006) INIT(' '), \*/  2 O\_BATCH\_ID CHAR (032) INIT(' '),  2 O\_PVV CHAR (004) INIT(' '),  2 O\_ERROR\_CDE CHAR (002) INIT(' '),  2 O\_CARD CHAR (019) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | UCDEFM1:UPDATE EMBOS\_FILE\_MASTER FILE\_STATUS  UCDEFD1: MASK PVV  UCDEFD3: UPDATE EMBOS\_FILE\_DETAIL COND TO 5 AND PVV  SCDTRC:SELECT FROM KCCT.TRACK\_INFO  ICDTRC:INSERT INTO KCCT.TRACK\_INFO  SCDEFD3: SELECT FROM CRDT.EMBOS\_FILE\_MASTER & DETAIL |
| **Main Process Logic** |  |
| **PROGRAM MAIN LOOP** | THE MAIN PROCESSING STEPS OF THE PROGRAM ARE:   * GET CURRENT DATE * OPEN FILES (INPUT:**K14CPIN1** AND OUTPUT:**K14CPOU1)** AND GET THE FIRST RECORD * IN A LOOP TILL END OF INPUT FILE PERFORM THE FOLLOWING TASKS:   + RETRIEVE CARD DATA USING SERVICE SCDEFD3   + MOVE DATA TO OUTPUT AREA AND WRITE TO FILE:K14CPOU1   + UPDATE MASTER TABLE WITH STATUS=’S’ USING SERVICE UCDEFM1   + UPDATE TABLE DETAIL CONDITION & STATUS = 4 USING SERVICE UCDEFD1   + UPDATE TABLE DETAIL USING SERVICE UCDEFD3   + RETRIEVE TRACK DATA USING SERVICE SCDTRC   + INSERT INTO INSERT\_ROW\_TRACK\_INFO USING SERVICE ICDTRC   + COMMIT EVERY 10 ROWS   + READ NEXT INPUT RECORD * AT END, CLOSE INPUT & OUTPUT FILES |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | KCCT.TRACK\_INFO  CRDT.EMBOS\_FILE\_MASTER  CRDT.EMBOS\_FILE\_DETAIL |

#### 

#### [K15CP] - CREATES EMBOSSING FILE FOR SSIS AND CMOD FILE WITH THOSE CARDS NOT SENT TO PERSONALIZATION BUREAU

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | EXEC SQL DECLARE C1 CURSOR FOR  SELECT \*  FROM CRDT.EMBOS\_FILE\_DETAIL A, CRDT.EMBOS\_FILE\_MASTER B  WHERE A.FILE\_ID = B.FILE\_ID  AND A.KIND IN('NEW','PLA','MIG', 'RWL')  AND A.MATURITY\_DATE <= :WS\_DATE\_CUR  AND A.CONDITION = 5  AND CLASS\_PRODUCT = 'DC' |
| **OUTFILE1 FILE DECLARATION** | **K15CPOU1:**  DCL 1 REMBOS,  /\* EMBOSSING AREA LENGTH = 117 \*/  2 KINDCNT PIC '(06)9' INIT(0),  2 CC01 CHAR (001) INIT('{'),  2 CC02 CHAR (003) INIT('DCC'),  2 CC03 CHAR (001) INIT('{'),  2 CC04 CHAR (001) INIT('{'),  2 CC05 CHAR (003) INIT('EMB'),  2 CC06 CHAR (001) INIT('{'),  2 LINE1\_CARD CHAR (019) INIT(' '),  2 CC07 CHAR (001) INIT('}'),  2 FILLER01 CHAR (012) INIT(' '),  2 MM\_LHX CHAR (002) INIT(' '),  2 CC08 CHAR (001) INIT('/'),  2 YY\_LHX CHAR (002) INIT(' '),  2 FILLER02 CHAR (010) INIT(' '),  2 CC09 CHAR (001) INIT('}'),  2 HOLDER CHAR (026) INIT(' '),  2 CC10 CHAR (001) INIT('}'),  2 COMPANY CHAR (026) INIT(' '),  /\* INDENT AREA LENGTH = 26 \*/  2 CC11 CHAR (001) INIT('{'),  2 CC12 CHAR (003) INIT('OCR'),  2 CC13 CHAR (001) INIT('{'),  2 FILLER03 CHAR (012) INIT(' '),  2 CARD\_4LAST PIC '9999' ,  2 FILLER04 CHAR (001) INIT(' '),  2 CVV2\_INDENT PIC '999' ,  2 FILLER05 CHAR (001) INIT(' '),  /\* TRACK-1 AREA LENGTH = 49 \*/  2 CC14 CHAR (001) INIT('{'),  2 CC15 CHAR (003) INIT('ENC'),  2 CC16 CHAR (003) INIT('{%B'),  2 TRACK1\_CARD CHAR (016) INIT(' '),  2 CC17 CHAR (001) INIT('^'),  2 TRACK1\_NAME CHAR (026) ,  2 CC18 CHAR (001) INIT('^'),  2 YYMM\_LHXH\_1 PIC '9999' ,  2 SVC\_1 CHAR (003) INIT(' '),  2 CVV1\_1 PIC '999' ,  2 ZEROES\_10 CHAR (010) INIT('0000000000'),  2 CC19 CHAR (001) INIT('?'),  2 FILLER06 CHAR (001) INIT(' '),  /\* TRACK-2 AREA LENGTH =39 \*/  2 CC20 CHAR (001) INIT(';'),  2 TRACK2\_CARD CHAR (016) INIT(' '),  2 CC21 CHAR (001) INIT('='),  2 YYMM\_LHXH\_2 PIC '9999' ,  2 SVC\_2 CHAR (003) INIT(' '),  2 CVV1\_2 PIC '999' ,  2 CARD\_SEQ\_NUM PIC '99' ,  2 ZEROES\_8 CHAR (008) INIT('00000000'),  2 CC22 CHAR (001) INIT('?'),  /\* INSERTER AREA \*/  2 FILLER07 CHAR (001) INIT(' '),  2 CC23 CHAR (002) INIT('|1'),  /\*ΠΡΟΣΟΧΗ ΕΙΔΙΚΟΣ ΧΑΡ\*/  2 CC24 CHAR (002) INIT('\_0'),  2 CC25 CHAR (011) INIT('[4037 0883]'),  2 TEMPL\_NAME CHAR (012) INIT(' '),  2 CC26 CHAR (003) INIT('[1]'),  2 KINDCNT2 CHAR (006) ,  2 CC27 CHAR (003) INIT('[2]'),  2 BRANCH CHAR (003) ,  2 CC28 CHAR (003) INIT('[3]'),  2 SNAME CHAR (022) ,  2 CC29 CHAR (003) INIT('[4]'),  2 PNAME CHAR (001) ,  2 CC30 CHAR (003) INIT('[5]'),  2 FNAME CHAR (003) ,  2 CC31 CHAR (003) INIT('[6]'),  2 STREET CHAR (025) ,  2 CC32 CHAR (003) INIT('[7]'),  2 ZIPCODE CHAR (005) ,  2 CC33 CHAR (003) INIT('[8]'),  2 CITY CHAR (010) ,  2 CC34 CHAR (003) INIT('[9]'),  2 HMER CHAR (010) INIT(' '),  2 CC35 CHAR (003) INIT('[A]'),  /\* D. MEGREMIS @ 30.10.2017 - START \*/  2 POS\_LIMIT CHAR (005) ,  2 CC36 CHAR (003) INIT('[B]'),  2 WDRW\_LIMIT CHAR (005) ,  /\* D. MEGREMIS @ 30.10.2017 - END \*/  2 CC37 CHAR (003) INIT('[C]'),  2 FILE\_EMB\_NAM CHAR (003) INIT(' '),  2 FILE\_EMB\_CNT PIC '99999' ,  2 CC38 CHAR (003) INIT('[D]'),  2 MATRIX CHAR (015) INIT(' '),  2 CC39 CHAR (003) INIT('[K]'),  2 INS\_AR\_CARD CHAR (016) INIT(' '),  2 CC40 CHAR (003) INIT('[L]'),  2 INS\_AR\_DATE CHAR (010) INIT(' '),  2 CC41 CHAR (003) INIT('[M]'),  2 INS\_AR\_NAME CHAR (027) INIT(' '),  2 CC42 CHAR (003) INIT('[N]'),  /\* D. MEGREMIS @ 30.10.2017 - START \*/  2 INT\_LIMIT CHAR (005) ,  /\* A. REPOUSI @ 29.08.2017 - START \*/  2 CC43 CHAR (003) INIT('[R]'),  2 DOM\_CNP\_LIMIT CHAR (005) ,  2 CC44 CHAR (003) INIT('[S]'),  2 INT\_CNP\_LIMIT CHAR (005) ,  2 CC45 CHAR (003) INIT('[T]'),  2 DOM\_WTH\_LIMIT CHAR (005) ,  2 CC46 CHAR (003) INIT('[U]'),  2 INT\_WTH\_LIMIT CHAR (005) ,  /\* A. REPOUSI @ 29.08.2017 - END \*/  /\* D. MEGREMIS @ 30.10.2017 - END \*/  /\* USER DATA \*/  2 ICVV\_USER PIC '999' ,  2 PIN\_BLOCK CHAR (016) INIT(' '),  2 FILLER11 CHAR (001) INIT(' '),  2 PLS\_DEL\_TYPE CHAR (001) INIT(' '),  2 CC48 CHAR (003) INIT('{F{'),  2 CC49 CHAR (003) INIT('@@@'),  2 RWL\_MIG\_FLAG CHAR (001) INIT(' ');  **K15CPOU2:**  DCL 1 REMBOSVI,  /\* EMBOSSING AREA LENGTH = 117 \*/  2 KINDCNT PIC '(06)9' INIT(0),  2 CC01 CHAR (001) INIT('{'),  2 CC02 CHAR (003) INIT('DCC'),  2 CC03 CHAR (001) INIT('{'),  2 CC04 CHAR (001) INIT('{'),  2 CC05 CHAR (003) INIT('EMB'),  2 CC06 CHAR (001) INIT('{'),  2 LINE1\_CARD CHAR (019) INIT(' '),  2 CC07 CHAR (001) INIT('}'),  2 FILLER01 CHAR (012) INIT(' '),  2 MM\_LHX CHAR (002) INIT(' '),  2 CC08 CHAR (001) INIT('/'),  2 YY\_LHX CHAR (002) INIT(' '),  2 FILLER02 CHAR (010) INIT(' '),  2 CC09 CHAR (001) INIT('}'),  2 HOLDER CHAR (026) INIT(' '),  2 CC10 CHAR (001) INIT('}'),  2 COMPANY CHAR (026) INIT(' '),  /\* INDENT AREA LENGTH = 26 \*/  2 CC11 CHAR (001) INIT('{'),  2 CC12 CHAR (003) INIT('OCR'),  2 CC13 CHAR (001) INIT('{'),  2 FILLER03 CHAR (012) INIT(' '),  2 CARD\_4LAST PIC '9999' ,  2 FILLER04 CHAR (001) INIT(' '),  2 CVV2\_INDENT PIC '999' ,  2 FILLER05 CHAR (001) INIT(' '),  /\* TRACK-1 AREA LENGTH = 84 \*/  2 CC14 CHAR (001) INIT('{'),  2 CC15 CHAR (003) INIT('ENC'),  2 CC16 CHAR (003) INIT('{%B'),  2 TRACK1\_CARD CHAR (016) INIT(' '),  2 CC17 CHAR (001) INIT('^'),  2 TRACK1\_NAME CHAR (026) ,  2 CC18 CHAR (001) INIT('^'),  2 YYMM\_LHXH\_1 PIC '9999' ,  2 SVC\_1 CHAR (003) INIT(' '),  2 DISCRETIONARY CHAR (013) INIT('0000000000000'),  2 ZEROES\_1 CHAR (002) INIT('00'),  2 CVV1\_1 PIC '999' ,  2 ZEROES\_2 CHAR (002) INIT('00'),  2 ACI CHAR (001) INIT('0'),  2 ZEROES\_3 CHAR (003) INIT('000'),  2 CC20 CHAR (001) INIT('?'),  2 FILLER07 CHAR (001) INIT(' '),  /\* TRACK-2 AREA LENGTH =39 \*/  2 CC21 CHAR (001) INIT(';'),  2 TRACK2\_CARD CHAR (016) INIT(' '),  2 CC22 CHAR (001) INIT('='),  2 YYMM\_LHXH\_2 PIC '9999' ,  2 SVC\_2 CHAR (003) INIT(' '),  2 CVV1\_2 PIC '999' ,  2 CARD\_SEQ\_NUM PIC '99' ,  2 ZEROES\_8 CHAR (008) INIT('00000000'),  2 CC23 CHAR (001) INIT('?'),  /\* INSERTER AREA \*/  2 FILLER08 CHAR (001) INIT(' '),  2 CC24 CHAR (002) INIT('|1'),  /\*ΠΡΟΣΟΧΗ ΕΙΔΙΚΟΣ ΧΑΡ\*/  2 CC25 CHAR (002) INIT('\_0'),  2 CC26 CHAR (011) INIT('[4037 0883]'),  2 TEMPL\_NAME CHAR (012) INIT(' '),  2 CC27 CHAR (003) INIT('[1]'),  2 KINDCNT2 CHAR (006) ,  2 CC28 CHAR (003) INIT('[2]'),  2 BRANCH CHAR (003) ,  2 CC29 CHAR (003) INIT('[3]'),  2 SNAME CHAR (022) ,  2 CC30 CHAR (003) INIT('[4]'),  2 PNAME CHAR (001) ,  2 CC31 CHAR (003) INIT('[5]'),  2 FNAME CHAR (003) ,  2 CC32 CHAR (003) INIT('[6]'),  2 STREET CHAR (025) ,  2 CC33 CHAR (003) INIT('[7]'),  2 ZIPCODE CHAR (005) ,  2 CC34 CHAR (003) INIT('[8]'),  2 CITY CHAR (010) ,  2 CC35 CHAR (003) INIT('[9]'),  2 HMER CHAR (010) INIT(' '),  2 CC36 CHAR (003) INIT('[A]'),  2 POS\_LIMIT CHAR (005) ,  2 CC37 CHAR (003) INIT('[B]'),  2 WDRW\_LIMIT CHAR (005) ,  2 CC38 CHAR (003) INIT('[C]'),  2 FILE\_EMB\_NAM CHAR (003) INIT(' '),  2 FILE\_EMB\_CNT PIC '99999' ,  2 CC39 CHAR (003) INIT('[D]'),  2 MATRIX CHAR (015) INIT(' '),  2 CC40 CHAR (003) INIT('[K]'),  2 INS\_AR\_CARD CHAR (016) INIT(' '),  2 CC41 CHAR (003) INIT('[L]'),  2 INS\_AR\_DATE CHAR (010) INIT(' '),  2 CC42 CHAR (003) INIT('[M]'),  2 INS\_AR\_NAME CHAR (027) INIT(' '),  2 CC43 CHAR (003) INIT('[N]'),  2 INT\_LIMIT CHAR (005) ,  2 CC44 CHAR (003) INIT('[R]'),  2 DOM\_CNP\_LIMIT CHAR (005) ,  2 CC45 CHAR (003) INIT('[S]'),  2 INT\_CNP\_LIMIT CHAR (005) ,  2 CC46 CHAR (003) INIT('[T]'),  2 DOM\_WTH\_LIMIT CHAR (005) ,  2 CC47 CHAR (003) INIT('[U]'),  2 INT\_WTH\_LIMIT CHAR (005) ,  /\* USER DATA \*/  2 ICVV\_USER PIC '999' ,  2 PIN\_BLOCK CHAR (016) INIT(' '),  2 FILLER11 CHAR (001) INIT(' '),  2 PLS\_DEL\_TYPE CHAR (001) INIT(' '),  2 CC48 CHAR (003) INIT('{F{'),  2 CC49 CHAR (003) INIT('@@@'),  2 RWL\_MIG\_FLAG CHAR (001) INIT(' ');  **CMODFILE:**  DCL LINE CHAR (133) INIT(' ');  **K15CPOU3:**  DCL 1 REMBOALL,  2 KINDCNT PIC '(06)9' INIT(0),  2 HOLDER CHAR (026) INIT(' '),  2 COMPANY CHAR (026) INIT(' '),  2 TRACK1\_CARD CHAR (016) INIT(' '),  2 STREET CHAR (025) INIT(' '),  2 ZIPCODE CHAR (005) INIT(' '),  2 CITY CHAR (010) INIT(' '),  2 FILE\_EMB\_NAM CHAR (003) INIT(' '),  2 FILE\_EMB\_CNT PIC '99999' INIT(' '),  2 PLS\_DEL\_TYPE CHAR (001) INIT(' ');  **K15CPOU4:**  DCL 1 REMBOSPC,  /\* 1. EMBOSSING AREA (Emb) \*/  2 KINDCNT PIC '(06)9' INIT(0),  2 CC01 CHAR (001) INIT('{'),  2 CC02 CHAR (003) INIT('DCC'),  2 CC03 CHAR (001) INIT('{'),  2 CC04 CHAR (001) INIT('{'),  2 CC05 CHAR (003) INIT('EMB'),  2 CC06 CHAR (001) INIT('{'),  2 LINE1\_CARD CHAR (019) INIT(' '),  2 CC07 CHAR (001) INIT('}'),  2 FILLER01 CHAR (012) INIT(' '),  2 MM\_LHX CHAR (002) INIT(' '),  2 CC08 CHAR (001) INIT('/'),  2 YY\_LHX CHAR (002) INIT(' '),  2 FILLER02 CHAR (010) INIT(' '),  2 CC09 CHAR (001) INIT('}'),  2 HOLDER CHAR (026) INIT(' '),  2 CC10 CHAR (001) INIT('}'),  2 COMPANY CHAR (026) INIT(' '),  /\* 2. INDENT AREA (Ind) \*/  2 CC11 CHAR (001) INIT('{'),  2 CC12 CHAR (003) INIT('OCR'),  2 CC13 CHAR (001) INIT('{'),  2 FILLER03 CHAR (012) INIT(' '),  2 CARD\_4LAST PIC '9999' ,  2 FILLER04 CHAR (001) INIT(' '),  2 CVV2\_INDENT PIC '999' ,  2 FILLER05 CHAR (001) INIT(' '),  /\* 3. TRACK - 1 AREA (Tr1) \*/  2 CC14 CHAR (001) INIT('{'),  2 CC15 CHAR (003) INIT('ENC'),  2 CC16 CHAR (003) INIT('{%B'),  2 TRACK1\_CARD CHAR (016) INIT(' '),  2 CC17 CHAR (001) INIT('^'),  2 TRACK1\_NAME CHAR (026) ,  2 CC18 CHAR (001) INIT('^'),  2 YYMM\_LHXH\_1 PIC '9999' ,  2 SVC\_1 CHAR (003) INIT(' '),  2 CVV1\_1 PIC '999' ,  2 ZEROES\_10 CHAR (010) INIT('0000000000'),  2 CC19 CHAR (001) INIT('?'),  2 FILLER06 CHAR (001) INIT(' '),  /\* 4. TRACK - 2 AREA (Tr2) \*/  2 CC20 CHAR (001) INIT(';'),  2 TRACK2\_CARD CHAR (016) INIT(' '),  2 CC21 CHAR (001) INIT('='),  2 YYMM\_LHXH\_2 PIC '9999' ,  2 SVC\_2 CHAR (003) INIT(' '),  2 CVV1\_2 PIC '999' ,  2 CARD\_SEQ\_NUM PIC '99' ,  2 ZEROES\_8 CHAR (008) INIT('00000000'),  2 CC22 CHAR (001) INIT('?'),  /\* 5. INSERTER AREA (Ins) \*/  2 FILLER07 CHAR (001) INIT(' '),  2 CC23 CHAR (002) INIT('|1'),  2 CC24 CHAR (002) INIT('\_0'),  2 CC25 CHAR (011) INIT('[4037 0883]'),  2 TEMPL\_NAME CHAR (012) INIT(' '),  2 CC26 CHAR (003) INIT('[1]'),  2 KINDCNT2 CHAR (006) ,  2 CC27 CHAR (003) INIT('[2]'),  2 BRANCH CHAR (003) ,  2 CC28 CHAR (003) INIT('[3]'),  2 SNAME CHAR (022) ,  2 CC29 CHAR (003) INIT('[4]'),  2 PNAME CHAR (001) ,  2 CC30 CHAR (003) INIT('[5]'),  2 FNAME CHAR (003) ,  2 CC31 CHAR (003) INIT('[6]'),  2 STREET CHAR (025) ,  2 CC32 CHAR (003) INIT('[7]'),  2 ZIPCODE CHAR (005) ,  2 CC33 CHAR (003) INIT('[8]'),  2 CITY CHAR (010) ,  2 CC34 CHAR (003) INIT('[9]'),  2 HMER CHAR (010) INIT(' '),  2 CC35 CHAR (003) INIT('[A]'),  2 POS\_LIMIT /\* ? \*/ CHAR (009) ,  2 CC36 CHAR (003) INIT('[B]'),  2 WDRW\_LIMIT /\* ? \*/ CHAR (009) ,  2 CC37 CHAR (003) INIT('[C]'),  2 FILE\_EMB\_NAM /\* ? \*/ CHAR (003) INIT(' '),  2 FILE\_EMB\_CNT /\* ? \*/ PIC '99999' ,  2 CC38 CHAR (003) INIT('[D]'),  2 MATRIX /\* ? \*/ CHAR (016) INIT(' '),  2 CC39 CHAR (003) INIT('[K]'),  2 EMPLOYEE\_ID CHAR (016) INIT(' '),  2 CC40 CHAR (003) INIT('[L]'),  2 DEPARTMENT\_ID CHAR (010) INIT(' '),  2 CC41 CHAR (003) INIT('[M]'),  2 COMPANY\_NAME CHAR (027) INIT(' '),  2 CC42 CHAR (003) INIT('[N]'),  2 INT\_LIMIT /\* ? \*/ CHAR (009) ,  2 CC43 /\* ? \*/ CHAR (003) INIT(' '),  2 FILLER09 /\* ? \*/ CHAR (001) INIT(' '),  2 FILLER10 /\* ? \*/ CHAR (007) INIT(' '),  /\* 6. USER DATA (UsDa) \*/  2 ICVV\_USER PIC '999' ,  2 PIN\_BLOCK CHAR (016) INIT(' '),  2 FILLER11 CHAR (001) INIT(' '),  2 PLS\_DEL\_TYPE CHAR (001) INIT(' '),  2 FILLER12 CHAR (003) INIT(' '),  2 CC44 CHAR (005) INIT('{END{'),  2 CC45 CHAR (006) INIT('@@@@@@');  **K15CPOU5:**  DCL 1 AS400REC, /\* MACHINE FORMAT \*/  2 KINDCNT PIC'(06)9' INIT(0),/\* A/A ΕΓΓΡΑΦΗΣ ΑΡΧΕΙΟΥ \*/  2 CC01 CHAR(01) INIT('{'),  2 CC02 CHAR(03) INIT('DCC'),  2 CC03 CHAR(01) INIT('{'),  /\* ΜΠΡΟΣΤΙΝΗ ΠΛΕΥΡΑ ΚΑΡΤΑΣ \*/  2 CC04 CHAR(01) INIT('{'),  2 CC05 CHAR(03) INIT('EMB'),  /\* 1Η ΓΡΑΜΜΗ \*/  2 CC06 CHAR(01) INIT('{'),  2 LINE1\_CARD CHAR(19) INIT(' '), /\*CARD NUMBER WITH SPACE/4\*/  2 CC07 CHAR(01) INIT('}'),  /\* 2Η ΓΡΑΜΜΗ \*/  2 FILLER01 CHAR(12) INIT(' '),  2 MM\_LHX CHAR(02) INIT(' '), /\*PREPAID-ΣΙΤΙΣΗ SPACE\*/  2 CC08 CHAR(1) INIT('/'),  2 YY\_LHX CHAR(02) INIT(' '), /\*PREPAID-ΣΙΤΙΣΗ SPACE\*/  2 FILLER02 CHAR(10) INIT(' '),  2 CC09 CHAR(01) INIT('}'),  /\* 3Η ΓΡΑΜΜΗ ΓΙΑ ΦΥΣΙΚΑ ΠΡΟΣΩΠΑ\*/  2 HOLDER CHAR(26) INIT(' '), /\*EMBOSSING NAME\*/  2 CC10 CHAR(01) INIT('}'),  /\* 4Η ΓΡΑΜΜΗ ΓΙΑ ΝΟΜΙΚΑ ΠΡΟΣΩΠΑ\*/  2 COMPANY CHAR(26) INIT(' '), /\* SPACE ALWAYS \*/  /\* ΠΙΣΩ ΠΛΕΥΡΑ ΚΑΡΤΑΣ \*/  2 CC11 CHAR(01) INIT('{'), /\* INDENT AREA 118\*/  2 CC12 CHAR(03) INIT('OCR'),  2 CC13 CHAR(01) INIT('{'),  /\* 1Η ΓΡΑΜΜΗ\*/  2 FILLER03 CHAR(12) INIT(' '),  2 CARD\_4LAST PIC'9999', /\*LAST 4 DIGITS OF CARD\*/  2 FILLER04 CHAR(01) INIT(' '),  2 CVV2\_INDENT PIC'999', /\*TABLE FROM PFM \*/  2 FILLER05 CHAR(01) INIT(' '),  /\* ΜΑΓΝΗΤΙΚΗ ΠΙΣΤΑ\*/  /\* TRACK 1 \*/  2 CC14 CHAR(01) INIT('{'), /\* ARXH TRACK1 144\*/  2 CC15 CHAR(03) INIT('ENC'),  2 CC16 CHAR(03) INIT('{%B'),  2 TRACK1\_CARD CHAR(16) INIT(' '), /\* CARD NO 16 NORMAL \*/  2 CC17 CHAR(01) INIT('^'),  2 TRACK1\_NAME CHAR(26), /\* XENOGLWSA HOLDER \*/  2 CC18 CHAR(01) INIT('^'),  2 YYMM\_LHXH\_1 PIC'9999',  2 SVC\_1 CHAR(03) INIT(' '), /\*SERVICE CODE \*/  2 CVV1\_1 PIC'999',  2 ZEROES\_10 CHAR(10) INIT('0000000000'),  2 CC19 CHAR(01) INIT('?'),  2 FILLER06 CHAR(01) INIT(' '),  /\* TRACK 2\*/  2 CC20 CHAR(01) INIT(';'), /\* ; ARXH TRACK2 217\*/  2 TRACK2\_CARD CHAR(16) INIT(' '), /\* CARD NO 16 NORMAL \*/  2 CC21 CHAR(01) INIT('='),  2 YYMM\_LHXH\_2 PIC'9999',  2 SVC\_2 CHAR(03) INIT(' '), /\*SERVICE CODE \*/  2 CVV1\_2 PIC'999',  2 CARD\_SEQ\_NUM PIC'99' INIT('0'),  2 ZEROES\_8 CHAR(08) INIT('00000000'),  2 CC22 CHAR(01) INIT('?'), /\* ? TELOS TRACK2 \*/  2 FILLER07 CHAR(01) INIT(' '), /\* ARXH INSERTER AREA \*/  2 CC23 CHAR(002) INIT('|1'),  2 CC24 CHAR(02) INIT('\_0'),  2 CC25 CHAR(11) INIT('[4037 0883]'), /\*POSITION\*/  2 TEMPL\_NAME CHAR(12) INIT(' '), /\*FILE NAME\*/  2 CC26 CHAR(03) INIT('[1]'),  2 KINDCNT2 CHAR(06), /\* =AYXWN ARITHMOS 287\*/  2 CC27 CHAR(03) INIT('[2]'),  2 BRANCH CHAR(03), /\* FOR MASS 942 \*/  2 CC28 CHAR(03) INIT('[3]'),  2 SNAME CHAR(22),  2 CC29 CHAR(03) INIT('[4]'),  2 PNAME CHAR(01),  2 CC30 CHAR(03) INIT('[5]'),  2 FNAME CHAR(03),  2 CC31 CHAR(03) INIT('[6]'),  2 STREET CHAR(25), /\* FOR MASS FROM ARXEIO \*/  2 CC32 CHAR(03) INIT('[7]'),  2 ZIPCODE CHAR(05),  2 CC33 CHAR(03) INIT('[8]'),  2 CITY CHAR(10),  2 CC34 CHAR(03) INIT('[9]'),  2 HMER CHAR(10) INIT(' '), /\*NEXT WORKING DATE \*/  2 CC35 CHAR(03) INIT('[A]'),  2 POS\_LIMIT CHAR (009),  2 CC36 CHAR(03) INIT('[B]'),  2 WDRW\_LIMIT CHAR (009),  2 CC37 CHAR(03) INIT('[C]'),  2 FILE\_EMB\_NAM CHAR (003) INIT(' '),  2 FILE\_EMB\_CNT PIC '99999',  2 FILLER08 CHAR(19) INIT(' '), /\* TELOS INSERTER \*/  /\* CHIP DATA \*/  2 ICVV\_USER PIC'999', /\*USER DATA\*/  2 PIN\_BLOCK CHAR(16) INIT(' '), /\*FORMAT 5 \*/  /\* USER DATA \*/  2 OFFLINEPIN CHAR(01) INIT(' '), /\* ΣΥΜΠΛΗΡΩΝΕΤΑΙ ΑΠΟ AS/400 \*/  2 XLS CHAR(01) INIT(' '), /\* DPP ΑΠΟ AS/400 \*/  2 FILLER09 CHAR(03) INIT(' '),  2 PRODUCTCODE CHAR(03) INIT('029'), /\* = 029 FOR VOUCHER \*/  2 PROCCDCODE CHAR(02) INIT(' '),  2 IDENTITY CHAR(09) INIT(' '), /\* DATA GIA AS400 \*/  2 REQUEST\_DATE CHAR(10) INIT(' '),  2 FILLER10 CHAR(01) INIT(' '),  2 INT\_LIMIT CHAR (009),  2 MATRIX CHAR(15) INIT(' '), /\*ΠΡΟΣΟΧΗ ΗΤΑΝ PIC \*/  2 BIRTHDATE CHAR(10) INIT(' '),  2 I\_IP CHAR(10) INIT(' '),  2 COUNTRY CHAR(16) INIT(' '),  /\* MARK END OF CARD \*/  2 CC38 CHAR(05) INIT('{END{'),  2 CC39 CHAR(06) INIT('@@@@@@');  /\*ΓΡΑΜΜΟΓΡΑΦΗΣΗ ΓΙΑ 400\*/ |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | UCDEFD3:UPDATE CRDT.EMBOS\_FILE\_MASTER AND DETAIL  SCDEPM:SELECT FROM CRDT.EMBOS\_PARM  SCDECR:SELECT FROM CRDT.EMBOS\_COUNTER  UCDECR:UPDATE FROM CRDT.EMBOS\_COUNTER  SCCSPN1:SELECT FROM KCCT.SMARTPIN  UCCSPN1:UPDATE KCCT.SMARTPIN  UCDEFM2:CRDT.EMBOS\_FILE\_MASTER  UCCSPN:UPDATE KCCT.SMARTPIN  SRVIXAD:DELETE CARD - CLIENT RELATION  KCCSPN:DELETE FROM KCCT.SMARTPIN |
| **Main Process Logic** |  |
| **PROGRAM MAIN LOOP** | THE MAIN PROCESSING STEPS OF THE PROGRAM ARE:   * GET CURRENT DATE * OPEN OUTPUT FILES (K15CPOU1,CMODFILE,K15CPOU2,K15CPOU3,K15CPOU4,K15CPOU5) * OPEN CURSOR * FETCH FIRST ROW * IN A LOOP TILL END OF CURSOR PERFORM THE FOLLOWING TASKS:   + FOR CARDS WITH KIND NOT ‘PLA’ AND ‘RWL’ AND SMARTPIN INDICATOR = ‘Y’     - READ SMARTPIN USING SERVICE SCCSPN1     - UPDATE KCCT.SMARTPIN USING SERVICE UCCSPN1   + FOR CARDS WITH KIND ‘RWL’ AND ‘MIG’ (Recarding Visa to MC)     - READ SMARTPIN USING SERVICE SCCSPN1     - UPDATE KCCT.SMARTPIN USING SERVICE UCCSPN1   + COMPARE PREVIOUS WAY4 FILE NAME WITH CURRENT AND: IF PASSES (LYKOS\_STATUS\_CNT) > 0     - IF PASSES (LYKOS\_STATUS\_CNT) > 0 , UPDATE CRDT.EMBOS\_FILE\_MASTER TABLE WITH STATUS = ‘S’     - INITIALIZE LOCAL VARIABLES     - READ FILE NAME FROM CRDT.EMBOS\_PARM USING SERVICE SCDEPM     - GET LAST COUNTER USED BY INVOKING SERVICE SCDECR AND INCREASE IT BY 1     - UPDATE CRDT.EMBOS\_COUNTER WITH LAST COUNTER USED USING SERVICE UCDECR   + READ FILE NAME FROM CRDT.EMBOS\_PARM AGAIN ONLY FOR RECARDING     - IF PRODUCT\_GROUP ^= ‘04’ & DETAIL.KIND = 'RWL' READ FILENAME FROM CRDT.EMBOS\_PARM BY SERVICE SCDEPM   + MAINTAIN COUNTERS FOR LYKOS   + PROCEED TO EMBOSSING BASED ON:     - EVALUATING RECORD ID (VISA OR MC)     - CHECKING IF PROCESSED RECORD WILL BE WRITTEN TO THE FILE SENT TO LYKOS OR TO CMOD     - PERFORM MORE VALIDATIONS     - BASED ON THE RECIPIENT OF THE RECORD AS DETERMINED ABOVE FORMAT OUTPUT FILE ACCORDINGLY   + UPDATE CRDT.EMBOS\_FILE\_DETAIL (STATUS & CONDITION SET TO 5) WITH LYKOS FILE USING SERVICE UCDEFD3   + PRESERVE CURRENTLY FETCHED FILE NAME TO LOCAL VARIABLE FOR COMPARISONS AS SHOWN ABOVE   + FETCH NEXT ROW * IF NO MORE FETCHES (SQLCODE=100) :   + UPDATE MASTER TABLE CRDT.EMBOS\_FILE\_MASTER WITH LAST FETCH HELD IN MEMORY   + CLOSE ALL OUTPUT FILES AND RETURN |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  KCCT.SMARTPIN  CRDT.EMBOS\_FILE\_MASTER  CRDT.EMBOS\_COUNTER |

#### 

#### [K72CP] - BASED ON INPUT FILE VALIDATE ACCOUNTS AND EXPORT INFORMATION FOR PREPAID VOUCHER CARDS

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | **INPFILE1:**  DCL 1 INPUT\_DATA\_1,  /\*20230531 U714 - V.P START CODE\*/  5 I\_CARD CHAR(016),  /\*20230531 U714 - V.P END CODE\*/  5 I\_IBAN CHAR(027),  5 I\_LEGAL\_I\_IP CHAR(011); |
| **OUTFILE1 FILE DECLARATION** | **OUTFILE1**  DCL 1 OUTPUT\_DATA\_1,  /\*20230531 U714 - V.P START CODE\*/  5 O\_CARD CHAR(16) INIT(' '),  /\*20230531 U714 - V.P END CODE\*/  5 O\_IBAN CHAR(27) INIT(' '),  5 O\_LIP CHAR(11) INIT(' '),  5 O\_DEPOSIT\_ACC CHAR(16) INIT(' '),  5 O\_VALID\_STAT CHAR(01) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | SRVSIBA: RETRIEVE ACCOUNT FROM IBAN  SRVIXAE:FIND RELATION OF CLIENT WITH ACCOUNT |
| **Main Process Logic** |  |
| **PROGRAM MAIN LOOP** | THE MAIN PROCESSING STEPS OF THE PROGRAM ARE:   * GET CURRENT DATE * OPEN FILES (INPUT:**INPFILE1** AND OUTPUT:**OUTFILE1)** AND GET THE FIRST RECORD * IN A LOOP TILL END OF INPUT FILE PERFORM THE FOLLOWING TASKS:   + INVOKE SERVICE SRVSIBA USING IBAN TO GET ACCOUNT DATA   + SELECT VOUCHER DETAILS FROM CRDT.EMBOS\_FILE\_DETAIL BASED ON CARD NUMBER   + IF CONDITION = 5 SWITCH IT TO 1 ELSE EVALUATE REJECTION REASON AND:     - WHEN ‘B7’ STATUS =3     - WHEN ‘B8’ STATUS = 2     - WHEN ‘B9’ STATUS = 4     - OTHERWISE STATUS = 7   + WRITE OUTPUT FILE(OUTFILE1)   + READ NEXT RECORD * AT END OF INPUT CLOSE FILES AND RETURN |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### 

#### [K73CP] - CALCULATES COMMISSION FEES FOR RENEWALS PER IBAN AND FINDS THE AVAILABLE BALANCES - REPORTS TO THE CMOD OUTPUT FILE

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | **INPFILE1**  DCL 1 INPUT\_DATA\_1,  5 I\_CARD CHAR(019),  5 I\_PROD\_EVENT CHAR(004),  5 I\_PROD\_CODE CHAR(003),  5 I\_IBAN CHAR(027),  5 I\_LEGAL\_I\_IP CHAR(011),  5 I\_EMPL\_ID CHAR(020),  5 I\_DEPART\_ID CHAR(020),  5 I\_BARCODE CHAR(020),  /\*20230531 U714 - V.P START CODE\*/  5 I\_EXP\_DATE CHAR(010),  /\*20230531 U714 - V.P END CODE\*/  5 I\_DEPOSIT\_ACC CHAR(016),  5 I\_VALID\_STAT CHAR(001); |
| **OUTFILE1 FILE DECLARATION** | **OUTFILE1**  DCL 1 OUTPUT\_DATA\_1,  /\*20230531 U714 - V.P START CODE\*/  /\*5 O\_IBAN CHAR(027) INIT(' '),\*/  5 O\_LEGAL\_I\_IP CHAR(011) INIT(' '),  /\*20230531 U714 - V.P END CODE\*/  5 O\_SURNAME CHAR(040) INIT(' '),  5 O\_NAME CHAR(040) INIT(' '),  5 O\_FATHER\_NAME CHAR(040) INIT(' '),  /\*20230531 U714 - V.P START CODE\*/  5 O\_EXP\_DATE CHAR(010) INIT(' '),  /\*20230531 U714 - V.P END CODE\*/  5 O\_EMPL\_ID CHAR(020) INIT(' '),  5 O\_DEPART\_ID CHAR(010) INIT(' '),  5 O\_BARCODE CHAR(020) INIT(' '),  5 O\_MASKCARD CHAR(019) INIT(' '),  5 O\_RESPONSE\_CD CHAR(002) INIT(' '),  5 O\_REJECT\_RSN CHAR(086) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | TEMP\_CARDS(10000) CHAR(16); |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | SRVIIPS: FINDS SIDIPEL  SCCTCAL: DETRMINES COMMISION REQUEST  SRVIXAS:RETRIEVE PHYSICAL ENTITY'S SYDIPEL IP  SRVINMF:RETRIEVAL OF CLIENT’S MASTER DATA  SRVLAC5:RETRIEVAL OF DEBIT/CREDIT DEPOSIT ACCOUNTS  SRVCDES:PLACES BLOCK ON CLIENT DEPOSIT ACCOUNT  SRVAVBA: RETRIEVE AVAILABLE ACCOUNT BALANCE |
| **Main Process Logic** | CALCULATES COMMISSION FEES FOR RENEWALS PER IBAN AND FINDS THE AVAILABLE BALANCES - REPORTS TO THE CMOD OUTPUT FILE |
| **PROGRAM MAIN LOOP** | THE MAIN PROCESSING STEPS OF THE PROGRAM ARE:   * OPEN INPUT FILE:**INPFILE1** AND OUTPUT FILE **OUTFILE1** * GET CURRENT DATE * READ FIRST RECORD * INITIALIZE VARIABLES AND ASSIGN VALUE OF IBAN TO LOCAL VARIABLE L\_OLD\_IBAN * IN A LOOP TILL END OF INPUT FILE PERFORM THE FOLLOWING TASKS:   + IN AN INTERNAL LOOP AND AS LONG STAYS IN THE PROCESSING OF THE SAME BRANCH AND VALID\_STAT = ‘1’, , PERFORM THE FOLLOWING:     - CALCULATE FEES BY INVOKING SERVICE SCCTCAL     - RETRIEVES OWNER SIDIPEL BY INVOKING SERVICE SRVIXAS     - SUM UP COMMISSIONS/FEES     - WRITE OUTPUT FILE     - READ NEXT RECORD     - IF IT IS A NEW BRANCH FIND AVAILABLE BALANCE BY INVOKING SERVICE SRVAVBA * AT END OF INPUT FILE CLOSE FILES AND RETURN |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

#### 

#### [K16CP] - PROCESSES FILE WHICH WILL BE SENT TO SSIS AND THEN TO LYKOS

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | **K16CPI:**  DCL 1 REMBOS,  2 KINDCNT PIC '(06)9' INIT(0),  2 HOLDER CHAR (026) INIT(' '),  2 COMPANY CHAR (026) INIT(' '),  2 TRACK1\_CARD CHAR (016) INIT(' '),  2 STREET CHAR (025) INIT(' '),  2 ZIPCODE CHAR (005) INIT(' '),  2 CITY CHAR (010) INIT(' '),  2 FILE\_EMB\_NAM CHAR (003) INIT(' '),  2 FILE\_EMB\_CNT PIC '99999' INIT(' '),  2 PLS\_DEL\_TYPE CHAR (001) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | ICDTRC:RETRIEVE TRACK\_INFO  SCDEFD2:READ DETAILS TABLE  SCDTRC::INSERT\_ROW\_TRACK\_INFO |
| **Main Process Logic** |  |
| **PROGRAM MAIN LOOP** | THE MAIN PROCESSING STEPS OF THE PROGRAM ARE:   * GET CURRENT DATE * OPEN INPUT FILE:**K16CPI** AND GET THE FIRST RECORD * IN A LOOP TILL END OF INPUT FILE PERFORM THE FOLLOWING TASKS:   + INVOKE SERVICE SCDEFD2 TO RETRIEVE DETAILS FROM CRDT.EMBOS\_FILE\_DETAIL (CONDITION 5)   + MAINTAIN COUNTER FOR DEBITS   + FORMAT DATA FOR DEBIT   + RETRIEVE LATEST TRACK DATA   + PREPARE COMM AREA ICDTRC AND INVOKE ICDTRC TO INSERT DATA INTO TABLE TRACK\_INFO   + READ NEXT INPUT RECORD * AT END CLOSE FILE AND PRINT STATISTICS |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | TRACK\_INFO  CRDT.EMBOS\_FILE\_DETAIL |

#### 

#### [K54CP] - PROCESS EMBOSSING FILE AND THEN CRDT.EMBOS\_FILE\_DETAIL WITH CONDITION <5 OR >= 6. UPDATE THE REST FROM CONDITION 5 TO CONDITION=6

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | **K54CPIN:**  DCL 1 RK54CPIN,  2 FILLER1 CHAR(058) INIT(' '),  2 CARD CHAR(016) INIT(' '),  2 FILLER2 CHAR(049) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | PROCESS EMBOSSING FILE AND THEN CRDT.EMBOS\_FILE\_DETAIL WITH CONDITION <5 OR >= 6. UPDATE THE REST FROM CONDITION 5 TO CONDITION=6 |
| **PROGRAM MAIN LOOP** | THE MAIN PROCESSING STEPS OF THE PROGRAM ARE:   * GET CURRENT DATE * OPEN INPUT FILE:**K54CPIN** AND GET THE FIRST RECORD * IN A LOOP TILL END OF INPUT FILE PERFORM THE FOLLOWING TASKS:   + SELECT CARD FROM CRDT.EMBOS\_FILE\_DETAIL   + IF CONDITION < 5 OR >= 6 ISSUE ERROR   + FOR THE REST OF CONDITIONS UPDATE CRDT.EMBOS\_FILE\_DETAIL TO CONDITION/STATUS TO 6   + COMMIT PER ROW   + FETCH NEXT ROW * IF NO MORE ROWS CLOSE INPUT FILE AND RETURN |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### 

#### [K55CP] - RETRIEVE CARD DETAILS WITH CONDITION ‘8’/’9’ AND CREATE AN OUTPUT CMOD FILE

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | SELECT \*  FROM CRDT.EMBOS\_FILE\_DETAIL  WHERE KIND IN('NEW','PLA','PIN','MIG','RWL')  AND MATURITY\_DATE <= :WS\_DATE\_CUR  AND CONDITION IN (8,9)  AND PRODUCT\_GROUP <> '04'  AND CLASS\_PRODUCT = 'DC' |
| **OUTFILE1 FILE DECLARATION** | **K55CPOUT**  DCL FLINE CHAR(133) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | RETRIEVE CARD DETAILS WITH CONDITION ‘8’/’9’ AND CREATE AN OUTPUT CMOD FILE |
| **PROGRAM MAIN LOOP** | THE MAIN PROCESSING STEPS OF THE PROGRAM ARE:   * GET CURRENT DATE * OPEN CURSOR * OPEN OUTPUT FILE:**K55CPOUT** * FETCH FIRST ROW * IN A LOOP TILL END OF INPUT FILE PERFORM THE FOLLOWING TASKS:   + IF IT'S A NEW BRANCH PRINT HEADINGS   + PREPARE TO WRITE TO CMOD FILE   + LABEL CATEGORY VARIABLE BASED ON KIND AS:     - NEW: ΝΕΑ ΕΚΔΟΣΗ     - PIN: ΕΠΑΝΕΚΔΟΣΗ PIN     - PLA: ΕΠΑΝ/ΣΗ ΠΛΑΣΤΙΚΟΥ     - MIG:MIGRATION     - RWL:ΕΠΑΝ/ΣΗ ΛΟΓΩ ΛΗΞΗΣ     - FILL IN REST OF CRITICAL FIELDS     - IF MORE THAN 50 LINES WRITE HEADINGS     - WRITE OUTPUT FILE K55CPOUT   + FETCH NEXT ROW * IF NO MORE ROWS CLOSE OUTPUT FILE AND RETURN |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

#### 

#### [K90CP] - DAILY REPORT WITH PREPAID VOUCHER FOR CMOD

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | CURSOR ON CRDT.EMBOS\_FILE\_DETAIL WITH KIND IN('NEW','PLA','PIN','MIG','RWL') AND MATURITY DATE <= CURRENT DATE AND CONDITION 8 OR 9 AND PRODUCT GROUP = ‘04’ |
| **OUTFILE1 FILE DECLARATION** | **K90CPOUT**  DCL FLINE CHAR(133) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | OPENS CURSOR ON CRDT.EMBOS\_FILE\_DETAIL WITH CERTAIN CRITERIA (KIND) AND FOR EVERY ROW WRITES TO THE OUTPUT FILE FOR CMOD |
| **PROGRAM MAIN LOOP** | THE MAIN PROCESSING STEPS OF THE PROGRAM ARE:   * GET CURRENT DATE & TIME * OPEN CURSOR * OPEN OUTPUT FILE * FETCH FIRST * PRINT HEADINGS * IN A LOOP TILL END OF CURSOR PERFORM THE FOLLOWING TASKS:   + PREPARE CMOD OUTPUT AREA BY GIVING TITLES BASED ON CARD-KIND VALUE     - NEW → ΝΕΑ ΕΚΔΟΣΗ     - PIN → ΕΠΑΝΕΚΔΟΣΗ PIN     - PLA → ΕΠΑΝ/ΣΗ ΠΛΑΣΤΙΚΟΥ     - MIG → MIGRATION     - RWL → ΕΠΑΝ/ΣΗ ΛΟΓΩ ΛΗΞΗΣ     - TAKE CARE OF HEADINGS EVERY 50 LINES     - WRITE OUTPUT FILE K90CPOUT     - FETCH NEXT ROW * AT END CLOSE FILE K90CPOUT AND RETURN |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### 

#### [K41CP] - CREATE A DAILY STATISTICS FILE FOR LYKOS WITH DATA PER ISSUING BRANCH

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | **CURSORS**  **COUNT ISSUES BY BRANCH**  SELECT COUNT(\*)  ,ISSUING\_BRANCH  FROM CRDT.EMBOS\_FILE\_DETAIL  WHERE KIND IN ('MIG','NEW','PLA','RWL')  AND MATURITY\_DATE = :L\_DATE\_CUR  AND CONDITION = 6  AND PRODUCT\_CODE != '029' /\* NO CMOD FOR VOUCHER \*/  AND CLASS\_PRODUCT = 'DC'  **RETRIEVE DETAILS PER ISSUING BRANCH**  SELECT PRODUCT\_CODE,  KIND,  COUNT(\*)  FROM CRDT.EMBOS\_FILE\_DETAIL  WHERE KIND IN ('MIG','NEW','PLA','RWL')  AND MATURITY\_DATE = :L\_DATE\_CUR  AND CONDITION = 6  AND ISSUING\_BRANCH =: WS\_BRANCH  AND PRODUCT\_CODE != '029' /\* NO CMOD FOR VOUCHER \*/  AND CLASS\_PRODUCT = 'DC' |
| **OUTFILE1 FILE DECLARATION** | **OUTFILE1**  DCL ULINE CHAR (133) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | CREATES A REPORTS FILE WITH DATA CONCERNING THE ISSUANCE ACTIVITY PER BRANCH |
| **PROGRAM MAIN LOOP** | THE MAIN PROCESSING STEPS OF THE PROGRAM ARE:   * OPEN OUTPUT FILE OUTFILE1 * GET CURRENT DATE & TIME * OPEN CURSOR FOR DAILY BRANCH ISSUANCE EXCLUDING VOUCHER (‘029’) * FETCH FIRST ROW * IN A LOOP TILL END OF CURSOR PERFORM THE FOLLOWING TASKS:   + FIND BRANCH TITLE   + INITIALIZE BRANCH COUNTERS   + GET BRANCH DETAILS BY OPENING CURSOR WITH DETAILS PER ISSUING BRANCH AND FETCH FIRST ROW   + IN A LOOP TILL CURSOR END (100) PERFORM THE FOLLOWING TASKS:     - EVALUATE KIND AND PRODUCT AND ASSIGN TIMES PRODUCT WAS ISSUED (***THIS PROCESS COULD BE EASILY BE IMPLEMENTED BY ISSUING A CURSOR QUERY ON BRANCH, COUNT(\*) WITH OPTION GROUP BY PRODUCT***)     - PROCEED TO THE NEXT FETCH   + AT END OF INTERNAL CURSOR WRITE CMOD FILE FOR BRANCH   + ISSUE NEXT FETCH * CLOSE CURSORS * CLOSE FILE * PRINT STATISTICS |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### 

#### [K34CP] - OPEN CURSOR ON CRDT.EMBOS\_FILE\_DETAIL AND CREATE TWO OUTPUT FILES WITH REJECTED EMBOSSING CARDS FOR SSIS AND WAY4

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | CURSOR ON CRDT.EMBOS\_FILE\_DETAIL AND CRDT.EMBOS\_FILE\_MASTER AND STATUS IN ('8','9','10','14') AND CONDITION IN (8,9,10,14) AND MATURITY DATE <= CURRENT DATE AND PRODUCT ‘DC’ |
| **OUTFILE1 FILE DECLARATION** | **K34CPO**  DCL 1 RK34CPO,  2 O\_REC\_NO CHAR (06) INIT(' '),  2 O\_CARD\_EMB CHAR (26) INIT(' '),  2 O\_CARD\_NO CHAR (16) INIT(' '),  /\* 2 O\_FILLER CHAR (01) INIT(' '), \*/  2 O\_FILLER CHAR (02) INIT(' '),  2 O\_REJ\_CODE CHAR (03) INIT(' '),  2 O\_REJ\_DESCR CHAR (100) INIT(' '),  2 O\_FILE CHAR (15) INIT(' '),  2 O\_ELTA\_DATE CHAR (10) INIT(' '),  2 O\_ORIG\_REC\_ID CHAR (06) INIT(' '),  2 O\_EXCEPT\_FLG CHAR (01) INIT(' '),  2 O\_PLS\_DEL\_TYPE CHAR (02) INIT(' '),  2 WAY4\_FILE\_NAME CHAR (40) INIT(' '),  /\* 2 TRANSACTION\_ID PIC '(10)9' INIT(' '), \*/  2 TRANSACTION\_ID CHAR (255) INIT(' '),  /\* 2 BATCH\_ID CHAR (06) INIT(' '); \*/  2 BATCH\_ID CHAR (32) INIT(' ');  **K34CPO2**  DCL 1 RK34CPO2,  2 CARD CHAR (16) INIT(' '),  2 FILLER\_1 CHAR (01) INIT(';'),  2 SYDIPEL CHAR (11) INIT(' '),  2 FILLER\_2 CHAR (01) INIT(';'),  2 CLASSIFIER\_VALUE CHAR (01) INIT(' '),  2 FILLER\_3 CHAR (01) INIT(';'),  2 CLASSIFIER\_NAME CHAR (16) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | UCDEFD2:  SRVIXAS: |
| **Main Process Logic** |  |
| **PROGRAM MAIN LOOP** | THE MAIN PROCESSING STEPS OF THE PROGRAM ARE:   * GET CURRENT DATE & TIME * OPEN OUTPUT FILES (K34CPO, K34CPO2) * OPEN CURSOR FOR DAILY REJECTED CARDS * FETCH FIRST ROW * IN A LOOP TILL END OF CURSOR PERFORM THE FOLLOWING TASKS:   + FILL IN OUTPUT FIELDS WITH CURSOR VALUES   + WRITE TO OUTPUT FILE K34CPO   + IF CONDITION = 14 & REJECT\_REASON <> 'C5'     - FILL IN OUTPUT FIELDS WITH CURSOR VALUES     - WRITE TO OUTPUT FILE K34CPO2   + UPDATE CRDT.EMBOS\_FILE\_DETAIL USING SERVICE UCDEFD2 WITH STATUS & CONDITION = 11   + FETCH NEXT ROW * AT END OF CURSOR   + CLOSE CURSOR   + CLOSE FILES K34CPO/K34CPO2   + RETURN |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### 

#### [K37CP] - READS OUTGOING FILE THAT WILL BE SENT TO LYKOS AND CREATES A FILE WITH UNIQUE MAILER CODE TO BE SENT FOR UPDATES TO WAY4

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | **K37CPI**  DCL 1 I\_REC,  2 KINDCNT PIC '(06)9' INIT(0),  2 HOLDER CHAR (026) INIT(' '),  2 COMPANY CHAR (026) INIT(' '),  2 TRACK1\_CARD CHAR (016) INIT(' '),  2 STREET CHAR (025) INIT(' '),  2 ZIPCODE CHAR (005) INIT(' '),  2 CITY CHAR (010) INIT(' '),  2 FILE\_EMB\_NAM CHAR (003) INIT(' '),  2 FILE\_EMB\_CNT PIC '99999' INIT(' '),  2 PLS\_DEL\_TYPE CHAR (001) INIT(' '); |
| **OUTFILE1 FILE DECLARATION** | **K37CPO**  DCL 1 RK37CPO,  2 CARD\_NO CHAR(16) INIT(' '),  2 FILE CHAR(15) INIT(' '),  2 ORIG\_REC\_ID CHAR(06) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | READS OUTGOING FILE THAT WILL BE SENT TO LYKOS AND CREATE A FILE WITH UNIQUE MAILER CODE TO BE SENT FOR UPDATES TO WAY4 |
| **PROGRAM MAIN LOOP** | THE MAIN PROCESSING STEPS OF THE PROGRAM ARE:   * OPEN INPUT FILE **K37CPI** AND OUTPUT FILE **K37CPO** * GET THE FIRST RECORD * IN A LOOP TILL END OF INPUT FILE PERFORM THE FOLLOWING TASKS:   + POPULATE OUTPUT AREA FROM INPUT VARIABLES   + WRITE OUTPUT (K37CPO)   + MAINTAIN COUNTERS   + READ NEXT RECORD * AT END OF INPUT FILE   + CLOSE INPUT AND OUTPUT FILES   + PRINT STATISTICS |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

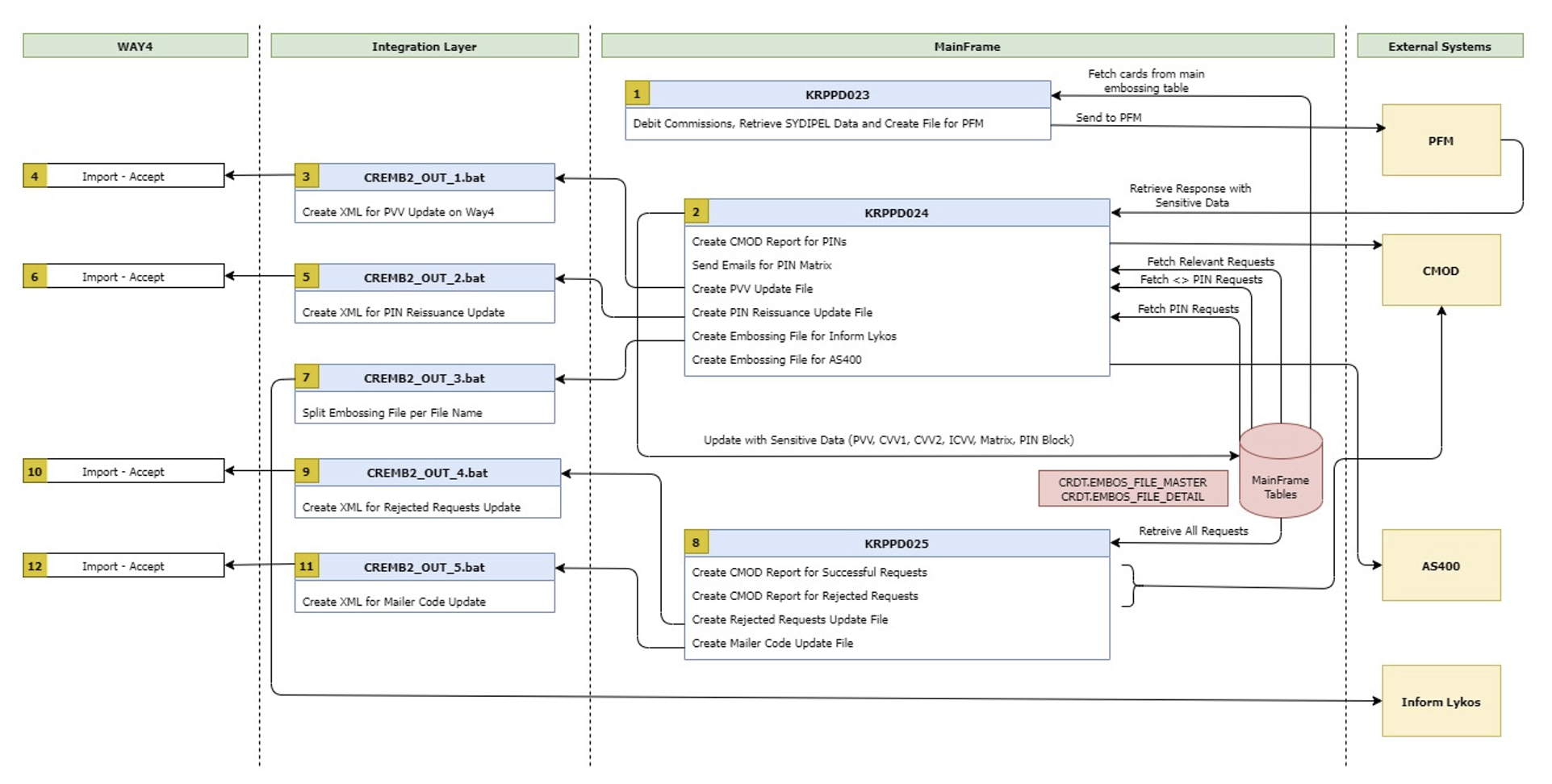
## 

## [Embossing - Prepaid]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Massive Loading Voucher |
| **Batch Flow Title** | KRPPD013-015 & KRPPD017-019 |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | Way4 |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   The prepaid embossing process uses client and cards tables to collect and validate the appropriate data that are required for the process. It sends a file to PFM for the cryptographic card data to be produced. It receives a file from PFM and delivers the embossing file to IL in order to be sent to Inform Lykos. |



### 

### Batch Integration Layer [Mainframe z/OS]

#### [KRPPD023] Insert Title

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| KRPPD023 | KF8DP |  |  | PREPAID: COMMISSIONS |
|  | KS77P |  |  | RETRIEVEMENT OF CARDHOLDER'S DETAILS FROM SYDIPEL |
|  | KY47P |  | CRDP.PD.PRKEA.CMOD.KAMPDPCA.SEQ | NON RECEIVABLES fees |
|  | CRD26 |  | CRDP.PD.PRKEA.CMOD.KAMPDPCA.SEQ | List NON RECEIVABLES fees from new and renewals VISA PREPAID cards |
|  | KS78P |  | CRDP.PD.PRKEA.SEMT.CRD02301.SEQ  CRDP.PD.PRKEA.SEMT.CRD02302.SEQ | Creation & sending of File to PFM |
|  | BACK01 | CRDP.PD.PRKEA.SEMT.CRD02301.SEQ | KATP.PD.PRKEA.GDGT.CRD02301.BACK | BACKUP THE FTOP FILES TO PFM |
|  | STEP00 | CRDP.PD.PRKEA.SEMT.CRD02301.SEQ | CRDP.PD.PRKEA.FTOP.CRD02301.SEQ | COPY PFM FILE WITHOUT HEADER |
|  | KS79P | CRDP.PD.PRKEA.SEMT.CRD02302.SEQ |  | Update CRDT.EMBOS\_FILE\_DETAIL & insertion on KCCT.TRACK\_INFO |
|  | KS80P | CRDP.PD.PRKEA.SEMT.CRD02301.SEQ | CRDP.PD.PRKEA.CMOD.CRD02301.SEQ | PRODUCTION OF A CMOD FILE FOR NEW ISSUANCES AND PLASTIC  REPLACEMENTS |
|  | STEP01 | CRDP.PD.PRKEA.SEMT.CRD02301.SEQ | CRDP.PD.PRKEA.INTT.CRD02304.SEQ | Creation of single entry file with the first line to send to PFM (actualy its name) |
|  | CRD22 | CRDP.PD.PRKEA.CMOD.CRD02301.SEQ |  | List from new and renewals  VISA PREPAID card per Name/Company name  - C M O D list |
|  | DEALLOC |  | CRDP.PD.PRKEA.SEMT.CRD02301.SEQ  CRDP.PD.PRKEA.SEMT.CRD02302.SEQ | DEALLOCATION OF FILES |

#### 

#### [KF8DP] Prepaid Commisiions

#### 

| **IL Process** | |
| --- | --- |
|  | |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | @SCCTSEL lookup of commission  @KCASEL  @KCPSEL  @SRVCOMW commission update to CSS for PREPAID KA WAY4  @SRVDDES,ACCOUNT Release  @SRVCDA9 Charge 9s Branch Store  @SRVCDAC  @SRVSACD PLACE BLOCK ON CLIENT DEPOSIT ACCOUNT  @SRVCDES COMMAREA WILL BE USED FOR SRVCDE1 CALL |
| **Main Process Logic** | ● Set current Date , Time  ● Create cursor on KCCT.COMMISSIONS\_TRN with STATUS='0' AND PMT\_METHOD='DESM' AND FREE\_IND='0' AND DATE(CREATED)>='01.10.2018' to get commissions  ● Loop of cursor described on next tab  ● Close cursor procedure |
| **PROGRAM MAIN LOOP** | * Initialize variables DEN\_XREWSE\_LOG , XREWSE\_9ARIA * Assign from cursor query return RKCCOMT on commissions the following string L\_CREATED = SUBSTR(RKCCOMT.CREATED,1,4)!!SUBSTR(RKCCOMT.CREATED,6,2)!! SUBSTR(RKCCOMT.CREATED,9,2); so from the sata returned the first 4 strings , followed by 2 string beginning on the 6 character followed from 2 strings from the ninth character * Invoike service SRVDDES FOR PARTIAL OR FULL RETENTION of Debit Account , When cursor input different than (RKCCOMT.BRANCH >='040' & RKCCOMT.BRANCH <='900' )assign @SRVDDES.IDATA.ATERM\_ID = ‘94299’ and @SRVDDES.IDATA.BRANCH\_SND ='942', If return of service <>0   + WHEN(22,31,33) - EKSWFLHMENOS LOGARIASMOS send message in console   + WHEN(21) non existing account , console message and assign to variable DEN\_XREWSE\_LOG = '1'   + Otherwise console message ,error for 'JOB CANCELLED' / DO\_ROLLBACK * IF DEN\_XREWSE\_LOG = FALSE (initialized value 0) then invoke service SRVCDAC and when cursor input for 6 first digits of cards SUBSTR(RKCCOMT.CARD,1,6) ='442317' assign @SRVCDAC.IDATA.OPERATION\_ID = '83911' ELSE @SRVCDAC.IDATA.OPERATION\_ID = '83666' and as above @SRVCDAC.IDATA.BRANCH\_SND ='942' if not between 040 and 900 and when the response other than 0   + WHEN(21,22,42,45,41,83,74) assign DEN\_XREWSE\_LOG = '1' and console message   + Otherwise console message ,error for 'JOB CANCELLED' / DO\_ROLLBAC * If DEN\_XREWSE\_LOG = '1' (client not charged) assigned in the invocation of the previous services for specific cases then do   + L\_BRANCH=''   + Call table SELECT KAT\_SYN FROM KATT.SYNENO WHERE KAT\_LOG = SUBSTR(RKCCOMT.ACCOUNT,1,3) to take L\_BRANCH   + Invoke service @SRVCDA9 with @SRVCDA9.IDATA.ACCOUNT =L\_BRANCH!!'9999981' and when SUBSTR(RKCCOMT.CARD,1,6) ='442317' assign @SRVCDA9.IDATA.OPERATION\_ID = '83911' ELSE @SRVCDA9.IDATA.OPERATION\_ID = '83666' and @SRVCDA9.IDATA.BRANCH\_SND ='942' if not between 040 and 900 and when the response is 0 XREWSE\_9ARIA ='1' otherwise console message and error   + Invoke service @SRVCDES TO PLACE A BLOCK ON ACCOUNT and if response <> 0 console message and error      * IF(XREWSE\_9ARIA = '1' then update STATUS field with 9 for the ID = RKCCOMT.ID otherwise UPD\_STATUS = 'P' for paid CCS commissions * Commit , Read next cursor line |
| **Arrays filled with corresponding values** | N/A |
|  | N/A |
| **CMS TABLES** | KCCT.COMMISSIONS\_TRN |

#### 

#### [KS77P] RETRIEVEMENT OF CARDHOLDER'S DETAILS FROM SYDIPEL

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | @SRVIXAS  @SRVILOC  @SRVIADR retrieve addresses  @SRVINMF retrieve Names  @PMNFKEA retrieve PMNT.KEADIK  @SRVINDS  @SRVIIPS |
| **Main Process Logic** | In order to Retrieve CARDHOLDER'S DETAILS FROM SYDIPEL we do the following   * Get CURRENT\_DATE * Open Cursor CURW select specific fields from CRDT.EMBOS\_FILE\_DETAIL WHERE CONDITION IN (0,8) and PRODUCT\_CODE IN ('143','144','145','146') AND MATURITY\_DATE <= :WS\_DATE\_CUR * Read first Line and execute next paragraph loop for each line , increase counter WS\_FETCH\_ROWS * Print outlist Statistics to Console |
| **PROGRAM MAIN LOOP** | * IF KIND = 'RWL' & PRODUCTION\_EVENT = 'RCRD' & PRODUCT\_CODE in ('143' , '144')) from CRDT.EMBOS\_FILE\_DETAIL we have issue and we update c with REJECT\_REASON = 'R1' , REJECT\_REASON\_TXT = 'ΛΟΓΩ RECARDING' and CONDITION = 13; * Else assign FOUND\_OK=0 and execute a procedure named SELECT\_STOIXEIA\_KRISI\_RTN which checks various conditions and takes actions based on the results of those checks. At the very end based on the results it updates like the previous step the CRDT.EMBOS\_FILE\_DETAIL table depending on the issue found. When there is Issue FOUND\_OK=1 and then update based on the code   IF (FOUND\_I\_IP=0) THEN CALL UPDATE\_ISSUES\_RTN(02);  ELSE IF (FOUND\_PERSON\_TYPE=0) THEN CALL UPDATE\_ISSUES\_RTN(03);  ELSE IF (FOUND\_NAMES=0) THEN CALL UPDATE\_ISSUES\_RTN(05);  ELSE IF (FOUND\_IDKEA=0) THEN CALL UPDATE\_ISSUES\_RTN(06);  ELSE IF (FOUND\_ADDRESS=0) THEN CALL UPDATE\_ISSUES\_RTN(04);  ELSE IF (BLOCKED=1) THEN CALL UPDATE\_ISSUES\_RTN(08);  ELSE IF (APAGOREYSH\_KINHSHS=1) THEN CALL UPDATE\_ISSUES\_RTN(09);  ELSE IF (THANONTAS=1) THEN CALL UPDATE\_ISSUES\_RTN(10);  ELSE IF (UNDERINV=1) THEN CALL UPDATE\_ISSUES\_RTN(11);   * If FOUND\_OK=0 update CRDT.EMBOS\_FILE\_DETAIL with REJECT\_REASON = ' '; REJECT\_REASON\_TXT = ' '; CONDITION= 1; and * Commit changes & and read Next cursor Line |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### [KY47P ] NEW ISSUE/RE-ISSUE VISA PREPAID ETE/KA CARD (NON-COLLECTABLE FEE)

| **IL Process** | |
| --- | --- |
|  | |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | Ν/Α |
| **INPFILE2 FILE DECLARATION** | Ν/Α |
| **OUTFILE1 FILE DECLARATION** | KY47POUT copy of FLINE with different DATA /\*RECORD LENGTH = 133\*/  From WRITE\_CMOD\_RTN example ,differentiation upon case  FLINE=' '!!WSC\_CNTREAD\_BRANCH!!(02)' '!!  RKCCOMT.CARD!!(02)' '!!  FSTR\_CUSTNAME!!(03)' '!!  WSC\_I\_IP!!(01)' '!!  WS\_CATEGORY!!(01)' '!!  'B7'!!  (01)' ' |
| **OUTFILE2 FILE DECLARATION** | Ν/Α |
| **INTERNAL TABLES/ARRAYS** | Ν/Α |
| **CRITICAL BOOLEAN INDICATORS** | Ν/Α |
| **SERVICES** | @SRVIXAS |
| **Main Process Logic** | * Get current date * Open cursor from commissions Table for specific cards where first 6 Digits in('442317','517179') and STATUS = '0' AND PMT\_METHOD = 'CASH' AND FREE\_IND = '0' AND (SUPPLY\_IND = 'ISSUE' OR SUPPLY\_IND = 'REISSUE') AND CURRENT DATE - 5 DAYS <= DATE(CREATED) **ORDER BY BRANCH** * Open output File KY47POUT * Console print header * Read first Cursor Line and check if should be part of embossing   + SELECT CRDT.EMBOS\_FILE\_DETAIL fetch first line from CRDT.EMBOS\_FILE\_DETAIL/ CRDT.EMBOS\_FILE\_DETAIL\_HIST WHERE CARD = :RKCCOMT.CARD AND CONDITION = 12 ORDER BY UPD\_TMSTMP DESC   + When found FLAG = '1' , increase counter WS\_SKIP   + When not found condition <>12 should be in the output file , FLAG = '0' , increase counter WS\_CNTREAD   + When FLAG = '1' read directly next fetch * Loop cursor as in next tab * Console print Statistics * Close output File KY47POUT |
| **PROGRAM MAIN LOOP** | * Increase counter WS\_CNTREAD\_BRANCH * For the first Branch read initialization page and A-A Branch ANALPAGE = 1, WS\_CNTREAD\_BRANCH = 1, write headings to output * Invoke service @SRVIXAS with RKCCOMT.CARD to confirm Sydipel connection * Write with procedure WRITE\_CMOD\_RTN the card to the output file   + For the case of number cards read ANALLIN>50 (next page) reprint headings   + Increase ANALLIN by 1   + For category RKCCOMT.SUPPLY\_IND in ( 'ISSUE' , 'REISSUE') include to the output line FLINE=' '!!WSC\_CNTREAD\_BRANCH!!(02)' '!! RKCCOMT.CARD!!(02)' '!!   FSTR\_CUSTNAME!!(03)' '!! WSC\_I\_IP!!(01)' '!! WS\_CATEGORY!!(01)' '!!'B7'!!(01)' '!!'NON-COLLECTED FEE'   * Continue with next line and do the checks as above if should be part of embossing   + SELECT CRDT.EMBOS\_FILE\_DETAIL fetch first line from CRDT.EMBOS\_FILE\_DETAIL/ CRDT.EMBOS\_FILE\_DETAIL\_HIST WHERE CARD = :RKCCOMT.CARD AND CONDITION = 12 ORDER BY UPD\_TMSTMP DESC   + When found FLAG = '1' , increase counter WS\_SKIP   + When not found condition <>12 should be in the output file , FLAG = '0' , increase counter WS\_CNTREAD   + When FLAG = '1' read directly next fetch |
| **Arrays filled with corresponding values** | Ν/Α |
| **CMS TABLES** | KCCT.COMMISSIONS\_TRN  CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_DETAIL\_HIST |

#### [KS78P]Sending File tp PFM

| **IL Process** | |
| --- | --- |
|  | |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | Ν/Α |
| **INPFILE2 FILE DECLARATION** | Ν/Α |
| **OUTFILE1 FILE DECLARATION** | DCL 1 RFTODPMS, /\* RECORD LENGTH 209 \*/  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME\_1 CHAR(30) INIT(' '), /\* ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΚΑΤΟΧΟΥ \*/  2 CUSTNAME\_2 CHAR(30) INIT(' '),  2 EXP\_DATE CHAR(04) INIT(' '), /\* ΗΜ/ΝΙΑ ΛΗΞΗΣ \*/  2 SVC CHAR(03) INIT(' '),  2 PVV CHAR(04) INIT(' '),  2 CVC2 CHAR(03) INIT(' '),  2 CVC1 CHAR(03) INIT(' '),  2 ADDRESS\_1 CHAR(30) INIT(' '), /\* ΔΙΕΥΘΥΜΣΗ- ΑΡΙΘΜΟΣ \*/  2 ADDRESS\_2 CHAR(30) INIT(' '), /\* TT-ΠΟΛΗΣ \*/  2 ACTION\_CODE CHAR(01) INIT(' '),  2 MAILER\_CODE CHAR(01) INIT(' '),  2 ENCRYPTED\_PIN CHAR(05) INIT(' '),  2 ERRORCODE CHAR(02) INIT(' '),  2 SPACE1 CHAR(01) INIT(' '),  2 SPACE2 CHAR(15) INIT(' '), /\* ΛΕΚΤΙΚΟ ΠΡΟΙΟΝΤΟΣ \*/  2 SPACE3 CHAR(15) INIT(' '), /\* CRA XXXXXXXXXXX \*/  2 SPACE4 CHAR(15) INIT(' '), /\* ΚΑΤΑΣΤΗΜΑ!!' '!!KIND \*/  2 E\_PIN CHAR(01) INIT('Y'); |
| **OUTFILE2 FILE DECLARATION** | DCL 1 RKS78PO2, /\* RECORD LENGTH 68 \*/  2 FILE\_ID PIC'(10)9' INIT(0),  2 BATCH\_ID CHAR(32) INIT(' '),  2 CARD CHAR(19) INIT(' '),  2 PRODUCTION\_EVENT CHAR(04) INIT(' '),  2 CONDITION PIC'99' INIT(0),  2 FLAG\_EPEX CHAR(01) INIT('0'); |
| **INTERNAL TABLES/ARRAYS** | Ν/Α |
| **CRITICAL BOOLEAN INDICATORS** | Ν/Α |
| **SERVICES** | @IDBCCFT |
| **Main Process Logic** | * Gets the current date * Retrieves the name of the file to process by invoking the service IDBCCFT * Opens the two files (KS78PO1 and KS78PO2) for writing * Writes the filename to KS78PO1 * Declare the cursor in Table CRDT.EMBOS\_FILE\_DETAIL WHERE CONDITION = 1 AND MATURITY\_DATE <= :WS\_DATE\_CUR and AND PRODUCT\_CODE IN ('143','144','145','146') * Fetch Data and Begin loop as next tab * Closes the cursor after processing * Updates the file transaction log KCCT.FILE\_TRAN by SET NBOFTXS=:RKCCFTRN.NBOFTXS WHERE APPL =:RKCCFTRN.APPL   AND SUB\_APPL=:RKCCFTRN.SUB\_APPL AND NAME =:RKCCFTRN.NAME   * Closes the two files (KS78PO1 and KS78PO2) * Prints statistics |
| **PROGRAM MAIN LOOP** | Processes each fetched row within a loop   * calls WRITE\_PMS\_RTN and fill with code the RFTODPMS input * calls WRITE\_OUT2 with FILE\_ID , BATCH\_ID , CARD PRODUCTION\_EVENT , CONDITION , FLAG\_EPEX * Increases counters CDXG , FILE\_LINES for statistics |
| **Arrays filled with corresponding values** | Ν/Α |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  KCCT.FILE\_TRAN |

#### [KS79P]UPDATE CRDT.EMBOS\_FILE\_DETAIL and insert to KCCT.TRACK\_INFO

#### 

| **IL Process** | |
| --- | --- |
|  | |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATI**  **ON** | DCL 1 RKS79PU1, /\* RECORD LENGTH 68 \*/  2 FILE\_ID PIC'(10)9',  2 BATCH\_ID CHAR(32) ,  2 CARD CHAR(19) ,  2 PRODUCTION\_EVENT CHAR(04) ,  2 CONDITION PIC'99' ,  2 FLAG\_EPEX CHAR(01) ; |
| **INPFILE2 FILE DECLARATION** | Ν/Α |
| **OUTFILE1 FILE DECLARATION** | Ν/Α |
| **OUTFILE2 FILE DECLARATION** | Ν/Α |
| **INTERNAL TABLES/ARRAYS** | Ν/Α |
| **CRITICAL BOOLEAN INDICATORS** | Ν/Α |
| **SERVICES** | @ICDTRC INSERT KCCT.TRACK\_INFO  @SCDTRC SELECT KCCT.TRACK\_INFO |
| **Main Process Logic** | * Gets the current date * OPEN FILE(KS79PU1)for processing * The DO WHILE loop processes each record. And for each record where FLAG\_EPEX is '0' from input file, it updates the database, inserts new tracking information, and writes back the record. * After processing CLOSE FILE(KS79PU1) * Prints statistics of program |
| **PROGRAM MAIN LOOP** | for each record where FLAG\_EPEX is '0' :   * updates the database with UPDATE CRDT.EMBOS\_FILE\_DETAIL   SET CONDITION = 2, REC\_STATUS = '2'  WHERE FILE\_ID = :L\_FILEID AND BATCH\_ID= :RKS79PU1.BATCH\_ID AND CARD = :RKS79PU1.CARD AND PRODUCTION\_EVENT= :RKS79PU1.PRODUCTION\_EVENT AND CONDITION = :L\_COND increase counter WS\_UPDATE\_ROWS and commit for successful update   * invoke service @ICDTRC to insert TRACK\_INFO , increase CNT\_INS\_TRACK * REWRITE FILE(KS79PU1) with FLAG\_EPEX = '1' processed   ; |
| **Arrays filled with corresponding values** | Ν/Α |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  KCCT.TRACK\_INFO |

#### 

#### [KS80P ] PRODUCTION OF A CMOD FILE FOR NEW ISSUANCES AN

| **IL Process** | |
| --- | --- |
|  | |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | DCL 1 RKS80PI1,  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME\_1 CHAR(30) INIT(' '), /\* ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΚΑΤΟΧΟΥ \*/  2 CUSTNAME\_2 CHAR(30) INIT(' '),  2 EXP\_DATE CHAR(04) INIT(' '), /\* ΗΜ/ΝΙΑ ΛΗΞΗΣ \*/  2 SVC CHAR(03) INIT(' '),  2 PVV CHAR(04) INIT(' '),  2 CVC2 CHAR(03) INIT(' '),  2 CVC1 CHAR(03) INIT(' '),  2 ADDRESS\_1 CHAR(30) INIT(' '), /\* ΔΙΕΥΘΥΜΣΗ- ΑΡΙΘΜΟΣ \*/  2 ADDRESS\_2 CHAR(30) INIT(' '), /\* TT-ΠΟΛΗΣ \*/  2 ACTION\_CODE CHAR(01) INIT(' '),  2 MAILER\_CODE CHAR(01) INIT(' '),  2 ENCRYPTED\_PIN CHAR(05) INIT(' '),  2 ERRORCODE CHAR(02) INIT(' '),  2 SPACE1 CHAR(01) INIT(' '),  2 SPACE2 CHAR(15) INIT(' '), /\* ΛΕΚΤΙΚΟ ΠΡΟΙΟΝΤΟΣ \*/  2 SPACE3 CHAR(15) INIT(' '), /\* CRA XXXXXXXXXXX \*/  2 SPACE4 CHAR(15) INIT(' '), /\* ΚΑΤΑΣΤΗΜΑ!!' '!!KIND \*/  2 E\_PIN CHAR(01) INIT('Y'); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | KS80PO1 FILE RECORD OUTPUT; /\*RECLEN = 133 \*/ |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | * Retrieve Current Date: * Open Files input KS80PI1 , output KS80PO1 files * READ FILE(KS80PI1) * Execute the following loop in the next tab * Close files KS80PI1 , KS80PO1 |
| **PROGRAM MAIN LOOP** | This loop reads each record from the input file, processes it based on its kind (using L\_KIND), and then performs various operations such as selecting details, writing records, and printing headings :   * Assign from input file L\_KIND = SUBSTR(RKS80PI1.SPACE4,7,3) * When L\_KIND in ('NEW', 'PLA', 'MIG') ,   + increase counter WS\_CNTREAD , WS\_CNTREAD\_BRANCH   + SELECT CARDS FROM CRDT.EMBOS\_FILE\_DETAIL WHERE CARD = :RKS80PI1.CARD   AND CONDITION = 2   * + For the case of new branch WS\_BRANCH = SUBSTR(RKS80PI1.SPACE4,1,5) ANALPAGE = 1; , WS\_CNTREAD\_BRANCH = 1 and print header * Write cmod output file with following events   WHEN('FCC') WS\_CATEGORY=' ΝΕΑ ΕΚΔΟΣΗ'; /\* REPLACEMENT FRAUD \*/  WHEN('INS') WS\_CATEGORY=' ΝΕΑ ΕΚΔΟΣΗ'; /\* INSTANT ISSUANCE \*/  WHEN('MIG') WS\_CATEGORY=' ΝΕΑ ΕΚΔΟΣΗ'; /\* RENEWAL MIGRATION \*/  WHEN('NEW') WS\_CATEGORY=' ΝΕΑ ΕΚΔΟΣΗ'; /\* NEW ISSUANCE \*/  WHEN('RWL') WS\_CATEGORY=' ΝΕΑ ΕΚΔΟΣΗ'; /\* RENEWAL EXPIRED AUTO \*/  WHEN('EXP') WS\_CATEGORY=' ΕΠΑΝΕΚΔΟΣΗ'; /\* RENEWAL EXPIRED MANU \*/  WHEN('FRD') WS\_CATEGORY=' ΕΠΑΝΕΚΔΟΣΗ'; /\* RENEWAL FRAUD \*/  WHEN('LST') WS\_CATEGORY=' ΕΠΑΝΕΚΔΟΣΗ'; /\* RENEWAL LOST \*/  WHEN('PLA') WS\_CATEGORY=' ΕΠΑΝΕΚΔΟΣΗ'; /\* RENEWAL PLASTIC \*/  WHEN('RDC') WS\_CATEGORY=' ΕΠΑΝΕΚΔΟΣΗ'; /\* RENEWAL DESTROYED \*/  WHEN('RSU') WS\_CATEGORY=' ΕΠΑΝΕΚΔΟΣΗ';/\*RENEWAL RESTRICTED USE \*/  WHEN('STL') WS\_CATEGORY=' ΕΠΑΝΕΚΔΟΣΗ'; /\* RENEWAL STOLEN \*/  WHEN('MAS') WS\_CATEGORY=' ΜΑΖΙΚΗ ΕΚΔΟΣΗ'; /\* MASSIVE ISSUANCE \*/  /\*U714 V.P ADDITION OF NEWR - @22.09.2023\*/  WHEN('NEWR') WS\_CATEGORY='ΛΟΓΩ RECARDING';  And line  FLINE=' '!!WSC\_CNTREAD\_BRANCH!!(02)' '!!  SUBSTR(RKS80PI1.CARD,1,4)!!'\*\*\*\*\*\*'!!  SUBSTR(RKS80PI1.CARD,11,6)!!(02)' '!!  RKS80PI1.CUSTNAME\_1!!(04)' '!!  SUBSTR(RKS80PI1.SPACE3,5,11)!!(03)' '!!  WS\_CATEGORY!!(02)' '!!  DCLEMBOS\_FILE\_DETAIL.PRODUCT\_NAME; |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER |

#### 

#### [KRPPD024] EMBOSSING OF PREPAID PRODUCTS ON WAY4

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| KRPPD024 | STEP01 | CRDP.PD.PRKEA.INTT.CRD02304.SEQ  CRDP.PD.PRKEA.FTIP.PREPDPFM.SEQ | CRDP.PD.PRKEA.SEMT.CRD02401 | COPY THE FILE WITH THE HEADER AND THE PFM FILE TO A NEW FILE |
|  | BACK00 | CRDP.PD.PRKEA.SEMT.CRD02401.SEQ | KATP.PD.PRKEA.GDGT.CRDPFM24.BACK | CREATE BACKUP FOR PFM FILE |
|  | KS81P | CRDP.PD.PRKEA.SEMT.CRD02401.SEQ |  | UPDATES THE TABLE EMBOS\_FILE\_DETAIL  AND SETS THE CONDITION FROM 2 TO 3 |
|  | STEP02 | CRDP.PD.PRKEA.SEMT.CRD02401.SEQ | CRDP.PD.PRKEA.SEMT.CRD02402.SEQ | Sort fil e FIELDS=(1,243,C' ') |
|  | KS82P | CRDP.PD.PRKEA.SEMT.CRD02402.SEQ |  | UPDATES THE TABLE EMBOS\_FILE\_DETAIL  AND SETS THE CONDITION FROM 3 TO 4 |
|  | KS83P |  |  | Check Non performed transactions |
|  | KS84P |  |  | Papyros file creation |
|  | KS85P |  |  | PROCESS ALL REQUESTS FOR SMARTPIN REISSUANCE |
|  | KS86P |  | CRDP.PD.PRKEA.SEMT.CRD02403.SEQ  CRDP.PD.PRKEA.SEMT.CRD02404.SEQ | FROM CRDT.EMBOS\_FILE\_DETAIL REQUESTS AND  WRITES TO FILE ONLY THOSE WITH "NEW","MAS","MIG" FOR THE UPDATE OF PVV OF THOSE CARDS IN WAY4 |
|  | STEP03 | CRDP.PD.PRKEA.SEMT.CRD02403.SEQ | CRDP.PD.PRKEA.FTOP.CRD02400 | Sort file FIELDS=(C'FD',1,23,21X) |
|  | KS87P |  | CRDP.PD.PRKEA.SEMT.CRD02404.SEQ | SELECTS specific datA FROM CRDT.EMBOS\_FILE\_DETAIL |
|  | KS88P |  | CRDP.PD.PRKEA.SEMT.CRD02405.SEQ  CRDP.PD.PRKEA.SEMT.CRD02406.SEQ | CREATE FILE WITH ALL PIN REISSUANCE REQUESTS |
|  | STEP04 | CRDP.PD.PRKEA.SEMT.CRD02405.SEQ | CRDP.PD.PRKEA.FTOP.CRD02401.SEQ | Sort file FIELDS=(C'FD',1,352) |
|  | KS89P | CRDP.PD.PRKEA.SEMT.CRD02406.SEQ |  | UPDATE CONDITION=12 CRDT.EMBOS\_FILE\_DETAIL  INSERT KCCT.TRACK\_INFO |
|  | KS90P |  | CRDP.PD.PRKEA.SEMT.CRD02407.SEQ  CRDP.PD.PRKEA.INTT.CRD02409.SEQ  CRDP.PD.PRKEA.SEMT.CRD02410.SEQ  CRDP.PD.PREMC.SEMT.CRD02404.SEQ  CRDP.PD.PRKEA.CMOD.CRD02403.SEQ | PRODUCTION OF EMBOSSING FILE TO SEND TO LYKOS |
|  | STEP05 | CRDP.PD.PRKEA.SEMT.CRD02407.SEQ | CRDP.PD.PRKEA.FTOP.CRD02402.SEQ | E M B O S S I N G F I L E  Sort file FIELDS FIELDS=(C'FD',1,580) |
|  | STEP06 | CRDP.PD.PREMC.SEMT.CRD02404.SEQ | CRDP.PD.PREMC.FTOP.CRD02404.SEQ | M C E M B O S S I N G F I L E  INSERT HEADER + TRAILER + IDENTITY FOR ROWS Sort file FIELDS FIELDS=(C'FD',1,569) |
|  | BACK01 | CRDP.PD.PRKEA.FTOP.CRD02400.SEQ | KATP.PD.PRKEA.GDGT.CRD02400.BACK | CREATE PVV UPDATE BACKUP FILE |
|  | BACK02 | CRDP.PD.PRKEA.FTOP.CRD02401.SEQ | KATP.PD.PRKEA.GDGT.CRD02401.BACK | CREATE PIN REISSUANCE BACKUP FILE |
|  | BACK03 | CRDP.PD.PRKEA.SEMT.CRD02410.SEQ | KATP.PD.PRKEA.GDGT.CRD02403.BACK | CREATE AS400 FILE BACKUP |
|  | BACK04 | CRDP.PD.PRKEA.FTOP.CRD02402.SEQ | KATP.PD.PRKEA.GDGT.CRD02402.BACK | CREATE EMBOSSING FILE BACKUP |
|  | BACK05 | CRDP.PD.PREMC.FTOP.CRD02404.SEQ | KATP.PD.PREMC.GDGT.CRD02404.BACK | CREATE EMBOSSING FILE BACKUP(MC) |
|  | REPRO1 | CRDP.PD.PRKEA.SEMT.CRD02410.SEQ | CRDP.PD.PRKEA.PERP.AS400PRE.SEQ | CREATE FILE FOR AS400 TOTAL EMBOSSING FILE |
|  | CRD25I |  | CRDP.PD.PRKEA.CMOD.CRD02403.SEQ | CMOD ERRORS |
|  | DEALLOC |  | CRDP.PD.PRKEA.SEMT.CRD02401.SEQ  CRDP.PD.PRKEA.SEMT.CRD02402.SEQ  CRDP.PD.PRKEA.SEMT.CRD02403.SEQ  CRDP.PD.PRKEA.SEMT.CRD02404.SEQ  CRDP.PD.PRKEA.SEMT.CRD02405.SEQ  CRDP.PD.PRKEA.SEMT.CRD02406.SEQ  CRDP.PD.PRKEA.SEMT.CRD02407.SEQ  CRDP.PD.PRKEA.INTT.CRD02304.SEQ  =CRDP.PD.PRKEA.SEMT.CRD02410.SEQ  CRDP.PD.PREMC.SEMT.CRD02404.SEQ | DEALLOCATION OF FILES |
|  | CLRSEQ1 | CRDP.PD.PRKEA.FTIP.PREPDPFM.SEQ |  | PREPAID: CLEAR PFM FILE |

#### 

#### [KS81P] PROGRAM PROCESSES FILES RETRIEVED FROM PFM

#### 

| **IL Process** | |
| --- | --- |
|  | |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | DCL 1 RKS81PI, /\*LENGTH:243\*/  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME\_1 CHAR(30) INIT(' '), /\* ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΚΑΤΟΧΟΥ \*/  2 CUSTNAME\_2 CHAR(30) INIT(' '),  2 EXP\_DATE CHAR(04) INIT(' '), /\* ΗΜ/ΝΙΑ ΛΗΞΗΣ \*/  2 SVC CHAR(03) INIT(' '),  2 PVV CHAR(04) INIT(' '),  2 CVC2 CHAR(03) INIT(' '),  2 CVC1 CHAR(03) INIT(' '),  2 ICVV CHAR(03) INIT(' '),  2 PINBLOCK CHAR(16) INIT(' '),  2 ADDRESS\_1 CHAR(30) INIT(' '), /\* ΔΙΕΥΘΥΜΣΗ- ΑΡΙΘΜΟΣ \*/  2 ADDRESS\_2 CHAR(30) INIT(' '), /\* TT-ΠΟΛΗΣ \*/  2 ACTION\_CODE CHAR(01) INIT(' '),  2 MAILER\_CODE CHAR(01) INIT(' '),  2 ENCRYPTED\_PIN CHAR(05) INIT(' '),  2 ERRORCODE CHAR(02) INIT(' '),  2 SPACE1 CHAR(01) INIT(' '),  2 SPACE2 CHAR(15) INIT(' '), /\* ΛΕΚΤΙΚΟ ΠΡΟΙΟΝΤΟΣ \*/  2 SPACE3 CHAR(15) INIT(' '), /\* CRA XXXXXXXXXXX \*/  2 SPACE4 CHAR(15) INIT(' '), /\* ΚΑΤΑΣΤΗΜΑ!!' '!!KIND \*/  2 E\_PIN CHAR(01) INIT('Y'),  2 MATRIX CHAR(15) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | * Get Current Date * Initializes conditions in the database to a specific state CONDITION = 2, WHERE VALUE IS 3 :   + For the entries SELECT \* FROM CRDT.EMBOS\_FILE\_DETAIL   WHERE KIND IN ('NEW','PLA','PIN','MIG','RWL') AND MATURITY\_DATE <= :WS\_DATE\_CUR AND CONDITION = 3 AND SMARTPIN = 'Y' AND PRODUCT\_CODE IN ('143','144', '145', '146') update fields CONDITION = 2  REC\_STATUS = 2  UPD\_TMSTMP = CURRENT TIMESTAMP   * Verifies the sum of records in the input file by reading the file and increase counter WS\_CNTREAD with the number in SELECT NBOFTXS INTO :RKCCFTRN.NBOFTXS   FROM KCCT.FILE\_TRAN WHERE APPL='PFM' AND SUB\_APPL='NEW2' AND NAME = :FLNAME otherwise error   * Opens the input file KS81PI from PFM and reads records. * In a loop, it processes each record by selecting details, performing checks, and updating records in the database as depicted in the next tab. * Closes the input file KS81PI file * checks totals to ensure consistency. |
| **PROGRAM MAIN LOOP** | * increase counter WS\_CNTREAD * Gets EMBOSSING DATA from sELECT \* INTO :DCLEMBOS\_FILE\_DETAIL FROM CRDT.EMBOS\_FILE\_DETAIL WHERE CARD = :RKS81PI.CARD ORDER BY MATURITY\_DATE DESC FETCH FIRST 1 ROW ONLY * Manage error cases otherwise L\_KIND = SUBSTR(RKS81PI.SPACE4,7,3) and proceed with the increase in counters WHEN('NEW') WS\_CNTNEW = WS\_CNTNEW + 1   WHEN('PIN') WS\_CNTPIN = WS\_CNTPIN + 1  WHEN('PLA') WS\_CNTPLA = WS\_CNTPLA + 1  WHEN('MIG') WS\_CNTMIG = WS\_CNTMIG + 1  WHEN('RWL') WS\_CNTRWL = WS\_CNTRWL + 1  Again as initialization update fields CONDITION = 3 REC\_STATUS = 3 using the same queries /procedures   * Read next input line |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER  KCCT.FILE\_TRAN |

#### 

#### [KS82P] UPDATES THE EMBOSS FILE DETAIL

| **IL Process** | |
| --- | --- |
|  | |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | DCL 1 RKS82PIN,  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME\_1 CHAR(30) INIT(' '),  2 CUSTNAME\_2 CHAR(30) INIT(' '),  2 EXP\_DATE CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '),  2 PVV CHAR(04) INIT(' '),  2 CVC2 CHAR(03) INIT(' '),  2 CVC1 CHAR(03) INIT(' '),  2 ICVV CHAR(03) INIT(' '),  2 PINBLOCK CHAR(16) INIT(' '),  2 ADDRESS\_1 CHAR(30) INIT(' '),  2 ADDRESS\_2 CHAR(30) INIT(' '),  2 ACTION\_CODE CHAR(01) INIT(' '),  2 MAILER\_CODE CHAR(01) INIT(' '),  2 ENCRYPTED\_PIN CHAR(05) INIT(' '),  2 ERRORCODE CHAR(02) INIT(' '),  2 SPACE1 CHAR(01) INIT(' '),  2 SPACE2 CHAR(15) INIT(' '),  2 SPACE3 CHAR(15) INIT(' '),  2 SPACE4 CHAR(15) INIT(' '),  2 E\_PIN CHAR(01) INIT(' '),  2 MATRIX CHAR(15) INIT(' '),  2 UPD\_FLG CHAR(01) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | * Get Current Date * Check for incomplete requests by calling CHECK\_ANEKPLHRWTA\_RTN   + SELECT IFNULL(COUNT(\*),0) INTO :LQCNT FROM CRDT.EMBOS\_FILE\_DETAIL   WHERE KIND IN ('NEW','PLA','PIN','MIG','RWL') AND MATURITY\_DATE = :WS\_DATE\_CUR AND CONDITION = 2 AND SMARTPIN = :RKS82PIN.E\_PIN   * + If found display appropriate message and initiate ROLLBACK with procedure SERROR * Open input file KS82PIN * Read header & first Data Line * Main loop execution * Close input File KS82PIN * Print total results in Console |
| **PROGRAM MAIN LOOP** | * For each line of Data and Card make the following checks   + Check card exists and CONDITION = 3 from CRDT.EMBOS\_FILE\_DETAIL ordering by FILE\_ID , MATURITY\_DATE Descending (first row only)   + Check input E\_PIN in ('Y' ,'N') otherwise stop error with wrong parameters * For the case that input ( E\_PIN = 'Y' and SPACE4 substring ( 7,4 ) is not 'PLA' or 'RWL’) OR ( E\_PIN = 'Y' and SPACE4 substring ( 7,4 ) = and DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT = 'MIG' then call procedure SELECT\_SMARTPIN\_TRACK\_RTN   + Where the card existence is checked inside CRDT.SMARTPIN\_TRACK for the thr firdt row by timestamp   + For SQLCODE = 100 non existence update input file field RKS82PIN.UPD\_FLG = '9' * RKS82PIN.UPD\_FLG = '9' skip row and read next one * When DCLEMBOS\_FILE\_DETAIL.CONDITION = 3 update CRDT.EMBOS\_FILE\_DETAIL * With SET CONDITION = 4   ,REC\_STATUS = '4'  ,PVV = :RKS82PIN.PVV  ,CVV1 = :WS\_CVV1  ,CVV2 = :WS\_CVV2  ,ICVV = :WS\_ICVV  ,PIN\_BLOCK = :RKS82PIN.PINBLOCK  ,UPD\_TMSTMP = CURRENT TIMESTAMP for this specific RKS82PIN.CARD and increase counter WS\_UPDATED   * When the substring of SUBSTR(RKS82PIN.SPACE4,7,4) is in ('NEW' , 'PIN' ,'MIG' ) execute procedure UPDATE\_SMARTPIN\_RTN() that is tables KCCT.SMARTPIN AND CRDT.SMARTPIN\_TRACK * When the substring of SUBSTR(RKS82PIN.SPACE4,7,4) is in ( 'PLA' , 'RWL') and if DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT = 'MIG' again UPDATE\_SMARTPIN\_RTN() * Update rewrite input file flag RKS82PIN.UPD\_FLG = '1' when value is ' ' * When DCLEMBOS\_FILE\_DETAIL.CONDITION <2 and > 4 call SERROR with the appropriate message and rollback of programm * Read next input |
| **Arrays filled with corresponding values** |  |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER  CRDT.SMARTPIN\_TRACK  KCCT.SMARTPIN |

#### [KS83P] Check of outstanding Transactions

| **IL Process** | |
| --- | --- |
|  | |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | PRM=&PA |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | @SRVDAYS |
| **Main Process Logic** | * Retrieve the current date * Call CHECK\_ANEKPLHRWTA\_RTN routine to check for incomplete requests * Exit program |
| **PROGRAM MAIN LOOP** | Inside the procedure CHECK\_ANEKPLHRWTA\_RTN   * Count from CRDT.EMBOS\_FILE\_DETAIL , KIND IN ('NEW','MIG','PIN','PLA','RWL') , MATURITY\_DATE = :WS\_DATE\_CUR ,CONDITION = 1 ,SMARTPIN = PA , CLASS\_PRODUCT = 'PC' * If count is 0 continue with embossing * Else if count <> 0 do the following   + Message to console to rerun the procedure   + Find next working Day, date after the next working day   + Call Prepaid Cursor with same attributes as above   + For each line of cursor update MATURITY DATE to next working Day   + Close cursor |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### [KS84P] Papyros File

| **IL Process** | |
| --- | --- |
|  | |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | DCL 1 RKS84POUT,  2 CARD CHAR (16) INIT(' '),  2 FIL01 CHAR (01) INIT(' '),  2 REQUEST CHAR (01) INIT(' '), /\* '1' REISSUE,  '3' NEW \*/  2 FIL02 CHAR (01) INIT(' '),  2 REQ\_DATE CHAR (10) INIT(' '),  2 FIL03 CHAR (01) INIT(' '),  2 SND\_DATE CHAR (10) INIT(' '),  2 FIL04 CHAR (01) INIT(' '),  2 HOLDER CHAR (26) INIT(' '),  2 FIL05 CHAR (01) INIT(' '),  2 CUST\_NAME CHAR (40) INIT(' '),  2 FIL06 CHAR (01) INIT(' '),  2 COMP\_NAME CHAR (40) INIT(' '),  2 FIL07 CHAR (01) INIT(' '),  2 CRA\_ADDR CHAR (40) INIT(' '),  2 FIL08 CHAR (01) INIT(' '),  2 CRA\_TK CHAR (40) INIT(' '),  2 FIL09 CHAR (01) INIT(' '),  2 CRA\_CITY CHAR (40) INIT(' '),  2 FIL10 CHAR (01) INIT(' '),  2 CRA\_COUNTRY CHAR (40) INIT(' '),  2 FIL11 CHAR (01) INIT(' '),  2 MATRIX CHAR (15) INIT(' '),  2 FIL12 CHAR (01) INIT(' '),  2 FLAG\_ENT CHAR (02) INIT('NO'), /\* ΕΝΘΕΤΟ \*/  2 FIL13 CHAR (01) INIT(' '),  2 FLAG\_LET CHAR (02) INIT('NO'), /\* ΕΠΙΣΤΟΛΗ \*/  2 FIL14 CHAR (01) INIT(' '),  2 CARD\_ORIGIN CHAR (02) INIT(' '), /\* CC,DC \*/  2 FIL15 CHAR (01) INIT(' '),  2 PRODUCT CHAR (09) INIT(' '),  2 FIL16 CHAR (01) INIT(' '),  2 PRODUCT\_NAME CHAR (30) INIT(' '),  2 FIL17 CHAR (01) INIT(' '),  2 CNT PIC '(6)9' INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | @SRVINMF  @SRVIXAL  @UCDSPT2  @SMRPIN |
| **Main Process Logic** | * Retrieve the current date * Open the output file KS84POUT. * Open the cursor , increase counter WS\_CNTREAD * Execute Main Loop as described in the next tab * Close Cursor , Output File KS84POUT * Print Statistics |
| **PROGRAM MAIN LOOP** |  Fetch records from the KCCT.SMARTPIN table with A.CARD\_ORIGIN = 'PC'  AND A.REQUEST = '1' AND A.FLG\_DELIVER IN ('0','1')  AND A.MATRIX ^= ' ' AND A.FLG = '2' ,  AND A.PRODUCT IN ('000000144','000000143','000000145','000000146') . and from joining KCCT.SMARTPIN A ,CRDT.EMBOS\_FILE\_DETAIL B  WHERE A.CARD = B.CARD and the only difference from above is AND A.REQUEST = '3' and B.CONDITION = 4 using READ\_SPIN\_RTN.   Continue processing while there are records to fetch (SQLCODE = 0).   For each fetched record:   * Update the KCCT.SMARTPIN table (UPDATE\_SMRTPN with sql). * Update the CRDT.SMARTPIN\_TRACK table (UPDATE\_SMARTPIN\_TRACK with service @UCDSPT2). * If the delivery type is not via branch (RKCCSPIN.FLG\_DELIVER ^= '0'), handle additional processing:   + - Select card details (SELECT\_DETAILS\_RTN from CRDT.EMBOS\_FILE\_DETAIL TABLE)).     - If no details found (SQLCODE = 100), call SELECT\_CRA\_RTN to collect customer data from @SRVIXAL , @SRVINMF     - Else If DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT <> 'INS', call SELECT\_CRA\_RTN to get customer Data and write to the CMOD file (WRITE\_CMOD\_RTN).    Fetch the next record using READ\_SPIN\_RTN. |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | KCCT.SMARTPIN  CRDT.SMARTPIN\_TRACK  CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER |

#### [Ks85P] PROCESS ALL REQUESTS FOR SMARTPIN REISSUANCE

| **IL Process** | |
| --- | --- |
|  | |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | @CROSMS |
| **Main Process Logic** | * Retrieve the current date * Open the cursor * Execute Main Loop as described in the next tab * Print Statistics to console |
| **PROGRAM MAIN LOOP** | * The main loop fetches records using the cursor   SELECT \*  FROM KCCT.SMARTPIN A, CRDT.EMBOS\_FILE\_DETAIL B  WHERE A.CARD = B.CARD AND A.CARD\_ORIGIN = 'PC' AND A.REQUEST = '3' AND A.FLG\_DELIVER = '2' AND A.FLG = '2' AND A.MATRIX ^= ' '  AND B.MATURITY\_DATE <= :WS\_DATE\_CUR  AND CONDITION = 4 AND CLASS\_PRODUCT = 'PC';   * For each row   + Invoce service @CROSMS to send SMARTPIN.EMAIL   + Increase counter WS\_CNTSEND   + UPDATE sql KCCT.SMARTPIN SET FLG = '1'   + UPDATE sql CRDT.SMARTPIN\_TRACK SET MATRIX\_SEND\_STATUS = '1', MATRIX\_SEND\_DATE = :WS\_DATE\_CUR   + Commit changes |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | KCCT.SMARTPIN  CRDT.SMARTPIN\_TRACK  CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER |

#### [KS86P] Gother requests to update to WAY4 from Emboss\_File

| **IL Process** | |
| --- | --- |
|  | |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | DCL 1 O\_PINRESPONSE,  2 O\_CARD CHAR(19) INIT(' '),  2 O\_PVV CHAR(04) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | DCL 1 O\_TEMPKEY,  2 FILE\_ID DEC FIXED(10,0) INIT(0),  2 SPACE1 CHAR(01) INIT(' '),  2 BATCH\_ID CHAR(32) INIT(' '),  2 SPACE2 CHAR(01) INIT(' '),  2 CARD CHAR(19) INIT(' '),  2 SPACE3 CHAR(01) INIT(' '),  2 PRODUCTION\_EVENT CHAR(04) INIT(' '),  2 SPACE4 CHAR(01) INIT(' '),  2 CONDITION DEC FIXED(02,0) INIT(0),  2 SPACE5 CHAR(01) INIT(' '),  2 FLAG CHAR(01) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | * Open Output Files KS86POU1 , KS86PTMP * Open cursor SELECT \* FROM CRDT.EMBOS\_FILE\_DETAIL WHERE KIND NOT IN ('PIN') AND CONDITION = 4 AND PRODUCT\_CODE IN ('143','144','145','146') AND SMARTPIN = PA * For each line where DCLEMBOS\_FILE\_DETAIL.KIND in ('NEW' , 'MIG','NEWR')   + WRITE TO OUTPUT FILE FOR PVV UPD   + WRITE TO TEMP FILE FOR COND UPD * Close cursor , Output files * Write program Statistics to console |
| **PROGRAM MAIN LOOP** | Described Above |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAI |

#### [KS87P] Update conditions for specific Conditions

| **IL Process** | |
| --- | --- |
|  | |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | DCL 1 O\_TEMPKEY,  2 FILE\_ID DEC FIXED(10,0) INIT(0),  2 SPACE1 CHAR(01) INIT(' '),  2 BATCH\_ID CHAR(32) INIT(' '),  2 SPACE2 CHAR(01) INIT(' '),  2 CARD CHAR(19) INIT(' '),  2 SPACE3 CHAR(01) INIT(' '),  2 PRODUCTION\_EVENT CHAR(04) INIT(' '),  2 SPACE4 CHAR(01) INIT(' '),  2 CONDITION DEC FIXED(2,0) INIT(0),  2 SPACE5 CHAR(01) INIT(' '),  2 FLAG CHAR(01) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | @UCDEFD1 UPDATE CONDITION AND PVV |
| **Main Process Logic** | * Open input File KS87PTMP * Read Input File * For each input Line   + If O\_TEMPKEY.CONDITION not '4' error   + Get Data from CRDT.EMBOS\_FILE\_DETAIL based on input file Data FILE\_ID ,BATCH\_ID, CARD ,PRODUCTION\_EVENT, CONDITION   + Update Data using service @UCDEFD1 and CONDITION = 5 ,STATUS = '5' ,increase counter WS\_UPDATED   + Commit and read next line   + Rewrite O\_TEMPKEY.FLAG = '1' to input file KS87PTMP to depict new changes * Close input file * Print statistics to Console |
| **PROGRAM MAIN LOOP** | N/A |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### [KS88P] CREATE FILE WITH ALL PIN REISSUANCE REQUESTS

| **IL Process** | |
| --- | --- |
|  | |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | DCL 1 PINRESPONSE, /\* RECORD LENGTH 352 \*/  2 O\_WAY4FILENAME CHAR (040) INIT(' '),  2 O\_TRANSACTION\_ID CHAR (255) INIT(' '),  2 O\_BATCH\_ID CHAR (032) INIT(' '),  2 O\_PVV CHAR (004) INIT(' '),  2 O\_ERROR\_CDE CHAR (002) INIT(' '),  2 O\_CARD CHAR (019) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | DCL 1 RKS78POU2, /\* RECORD LENGTH 64 \*/  2 FILE\_ID FIXED DEC(10,0) INIT(0),  2 BATCH\_ID CHAR(32) INIT(' '),  2 CARD CHAR(19) INIT(' '),  2 PRODUCTION\_EVENT CHAR(04) INIT(' '),  2 CONDITION FIXED DEC(02,0) INIT(0),  2 FLAG\_EPEX CHAR(01) INIT('0'); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | @SCDEFD3 |
| **Main Process Logic** | * Retrieve the current date * Open Output Files KS88POU1 , KS88POU2 * Open the cursor FROM CRDT.EMBOS\_FILE\_DETAIL WHERE CONDITION = 4   AND PRODUCT\_CODE IN ('143','144','145','146') AND KIND = 'PIN' meaning ONLY FOR PRODUCTION\_EVENT = PIN-PINF   * Execute Main Loop as described in the next tab * Close Cursor , Output Files KS88POU1 , KS88POU2 * Print Statistics to console |
| **PROGRAM MAIN LOOP** | For each fetch of cursor   * RETRIEVE CARD INFORMATION from service @SCDEFD3 * WRITE FILE(KS88POU1) from data acquired from @SCDEFD3 * WRITE FILE(KS88POU2) from data acquired from DCLEMBOS\_FILE\_DETAIL |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### [KS89P] UPDATE CONDITION to CRDT.EMBOS\_FILE\_DETAIL

| **IL Process** | |
| --- | --- |
|  | |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | DCL 1 RKS89PU1, /\* RECORD LENGTH 64 \*/  2 FILE\_ID FIXED DEC(10,0),  2 BATCH\_ID CHAR(32) ,  2 CARD CHAR(19) ,  2 PRODUCTION\_EVENT CHAR(04) ,  2 CONDITION FIXED DEC(02,0),  2 FLAG\_EPEX CHAR(01) ; |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | @UCDEFM1 UPDATE EMBOS\_FILE\_MASTER FILE\_STATUS  @UCDEFD1 MASK PVV  @UCDEFD3, UPDATE EMBOS\_FILE\_DETAIL COND TO 5 AND PVV  @SCDTRC SELECT FROM KCCT.TRACK\_INFO  @ICDTRC INSERT INTO KCCT.TRACK\_INFO  @SCDEFD3 SELECT FROM CRDT.EMBOS\_FILE\_MASTER AND CRDT.EMBOS\_FILE\_DETAIL |
| **Main Process Logic** | * Retrieve the current date * Open input File KS89PU1 * Read Input * Execute Main Loop as described in the next tab * Close input File KS89PU1 * Print Statistics to console |
| **PROGRAM MAIN LOOP** | For each line in the input where flag KS89PU1.FLAG\_EPEX = '0' do the following   * RETRIEVE CARD INFO using service @SCDEFD3 * UPDATE TABLE FILE\_MASTER using service @UCDEFM1 * UPDATE TABLE FILE\_DETAIL using service @UCDEFD1 * INSERT\_ROW\_TRACK\_INFO * REWRITE input FILE(KS89PU1) with flag RKS89PU1.FLAG\_EPEX = '1' , increase counter CWRITE1 |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER |

#### [KS90P] Creation of embossing file

#### 

| **IL Process** | |
| --- | --- |
|  | |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | DECLARE C1 CURSOR FOR  SELECT \*  FROM CRDT.EMBOS\_FILE\_DETAIL  WHERE KIND IN('NEW','PLA','PIN','MIG','RWL')  AND MATURITY\_DATE <= :WS\_DATE\_CUR  AND CONDITION IN (8,9)  AND PRODUCT\_GROUP = '04'  ORDER BY ISSUING\_BRANCH,CONDITION, KIND, CARD |
| **OUTFILE1 FILE DECLARATION** | **K90CPOUT**  DCL FLINE CHAR(133) INIT(' ');  **KS90POU1 : EMBOSSING FILE FOR SSIS (VISA)**  DCL 1 REMBOSV, /\* MACHINE FORMAT \*/  2 KINDCNT PIC'(06)9' INIT(0),/\* A/A ΕΓΓΡΑΦΗΣ ΑΡΧΕΙΟΥ \*/  2 CC01 CHAR(01) INIT('{'),  2 CC02 CHAR(03) INIT('DCC'),  2 CC03 CHAR(01) INIT('{'),  /\* ΜΠΡΟΣΤΙΝΗ ΠΛΕΥΡΑ ΚΑΡΤΑΣ \*/  2 CC04 CHAR(01) INIT('{'),  2 CC05 CHAR(03) INIT('EMB'),  /\* 1Η ΓΡΑΜΜΗ \*/  2 CC06 CHAR(01) INIT('{'),  2 CARD\_4NUMBER CHAR(19) INIT(' '), /\*CARD NUMBER WITH SPACE/4\*/  2 CC07 CHAR(01) INIT('}'),  /\* 2Η ΓΡΑΜΜΗ \*/  2 FILLER01 CHAR(12) INIT(' '),  2 MM\_LHX CHAR(02) INIT(' '), /\*PREPAID-ΣΙΤΙΣΗ SPACE\*/  2 CC08 CHAR(1) INIT('/'),  2 YY\_LHX CHAR(02) INIT(' '), /\*PREPAID-ΣΙΤΙΣΗ SPACE\*/  2 FILLER02 CHAR(10) INIT(' '),  2 CC09 CHAR(01) INIT('}'),  /\* 3Η ΓΡΑΜΜΗ ΓΙΑ ΦΥΣΙΚΑ ΠΡΟΣΩΠΑ\*/  2 HOLDER CHAR(26) INIT(' '), /\*EMBOSSING NAME\*/  2 CC10 CHAR(01) INIT('}'),  /\* 4Η ΓΡΑΜΜΗ ΓΙΑ ΝΟΜΙΚΑ ΠΡΟΣΩΠΑ\*/  2 COMPANY CHAR(26) INIT(' '), /\* SPACE ALWAYS \*/  /\* ΠΙΣΩ ΠΛΕΥΡΑ ΚΑΡΤΑΣ \*/  2 CC11 CHAR(01) INIT('{'), /\* INDENT AREA 118\*/  2 CC12 CHAR(03) INIT('OCR'),  2 CC13 CHAR(01) INIT('{'),  /\* 1Η ΓΡΑΜΜΗ\*/  2 FILLER03 CHAR(12) INIT(' '),  2 CARD\_4LAST PIC'9999', /\*LAST 4 DIGITS OF CARD\*/  2 FILLER04 CHAR(01) INIT(' '),  2 CVC2\_INDENT PIC'999', /\*TABLE FROM PFM \*/  2 FILLER05 CHAR(01) INIT(' '),  /\* ΜΑΓΝΗΤΙΚΗ ΠΙΣΤΑ\*/  /\* TRACK 1 \*/  2 CC14 CHAR(01) INIT('{'), /\* ARXH TRACK1 144\*/  2 CC15 CHAR(03) INIT('ENC'),  2 CC16 CHAR(03) INIT('{%B'),  2 TRACK1\_CARD CHAR(16) INIT(' '), /\* CARD NO 16 NORMAL \*/  2 CC17 CHAR(01) INIT('^'),  2 TRACK1\_NAME CHAR(26), /\* XENOGLWSA HOLDER \*/  2 CC18 CHAR(01) INIT('^'),  2 YYMM\_LHXH\_1 PIC'9999', /\*PREPAID-ΣΙΤΙΣΗ SPACE\*/  2 SVC\_1 CHAR(03) INIT(' '), /\*SERVICE CODE 226 \*/  /\* B.X. 19.11.2015 \*/  /\* FIELDS ONLY FOR VISA PREPAID \*/  2 DISCRET CHAR(13) INIT(' ') ,  2 ZEROES\_02\_V1 CHAR(02) INIT('00') ,  2 CVC1\_1 PIC'999' ,  2 ZEROES\_02\_V2 CHAR(02) INIT('00') ,  2 ACI CHAR(01) INIT(' ') ,  2 ZEROES\_03\_V1 CHAR(03) INIT('000'),  2 CC19 CHAR(01) INIT('?'),  2 FILLER06 CHAR(01) INIT(' '),  /\* TRACK 2\*/  2 CC20 CHAR(01) INIT(';'), /\* ; ARXH TRACK2 217\*/  2 TRACK2\_CARD CHAR(16) INIT(' '), /\* CARD NO 16 NORMAL \*/  2 CC21 CHAR(01) INIT('='),  2 YYMM\_LHXH\_2 PIC'9999', /\*PREPAID-ΣΙΤΙΣΗ SPACE\*/  2 SVC\_2 CHAR(03) INIT(' '), /\*SERVICE CODE 226 \*/  2 CVC1\_2 PIC'999',  2 CARD\_SEQ\_NUM PIC'99' INIT('0'),  2 ZEROES\_8 CHAR(08) INIT('00000000'),  2 CC22 CHAR(01) INIT('?'), /\* ? TELOS TRACK2 \*/  /\* CARD MAILER\*/  2 FILLER07 CHAR(01) INIT(' '), /\* ARXH INSERTER AREA \*/  /\* ΠΛΗΘΟΣ ΕΠΙΣΥΝΑΠΤΟΜΕΝΩΝ ΚΑΡΤΩΝ ΣΤΗΝ ΕΠΙΣΤΟΛΗ \*/  2 CC23\_1 CHAR(01) INIT('6A'X),  2 CC23\_2 CHAR(01) INIT('1'),  2 CC24 CHAR(02) INIT('\_0'),  /\* ΠΛΑΤΟΣ-ΥΨΟΣ ΕΠΙΣΥΝΑΠΤΟΜΕΝΩΝ ΚΑΡΤΩΝ ΣΤΗΝ ΕΠΙΣΤΟΛΗ \*/  2 CC25 CHAR(11) INIT('[4037 0883]'), /\*POSITION\*/  /\* ΚΑΘΟΡΙΖΕΙ ΤΗΝ ΜΗΤΡΑ ΤΗΣ DATACARD ΚΑΙ ΤΟ ΠΡΟΙΟΝ \*/  2 TEMPL\_NAME CHAR(12) INIT(' '), /\*FILE NAME\*/  /\* ΔΕΔΟΜΕΝΑ ΕΚΤΥΠΩΣΗΣ CARD MAILER \*/  2 CC26 CHAR(03) INIT('[1]'),  2 KINDCNT2 CHAR(06), /\* =AYXWN ARITHMOS 287\*/  2 CC27 CHAR(03) INIT('[2]'),  2 BRANCH CHAR(03), /\* FOR MASS 942 \*/  2 CC28 CHAR(03) INIT('[3]'),  2 SNAME CHAR(22),  2 CC29 CHAR(03) INIT('[4]'),  2 PNAME CHAR(01),  2 CC30 CHAR(03) INIT('[5]'),  2 FNAME CHAR(03),  2 CC31 CHAR(03) INIT('[6]'),  2 STREET CHAR(25), /\* FOR MASS FROM ARXEIO \*/  2 CC32 CHAR(03) INIT('[7]'),  2 ZIPCODE CHAR(05),  2 CC33 CHAR(03) INIT('[8]'),  2 CITY CHAR(10),  2 CC34 CHAR(03) INIT('[9]'),  2 INSERT\_DATE CHAR(10) INIT(' '), /\*NEXT WORKING DATE\*/  2 CC35 CHAR(03) INIT('[A]'),  2 POS\_LIMIT PIC'ZZ.ZZ9V,99', /\* 0 FOR ΣΙΤΙΣΗ \*/  2 CC36 CHAR(03) INIT('[B]'),  2 WDRW\_LIMIT PIC'ZZ.ZZ9V,99', /\* 0 FOR ΣΙΤΙΣΗ \*/  2 CC37 CHAR(03) INIT('[C]'),  2 FILE\_EMB\_NAM CHAR(08) INIT(' '), /\* GEMIZEI ME TO FILE NAME \*/  /\*ΓΡΑΜΜΟΓΡΑΦΗΣΗ ΓΙΑ LYKOS\*/  2 CC38 CHAR(03) INIT('[D]'),  2 SMART\_PIN CHAR(16) INIT(' '), /\* SMART PIN - BAR CODE \*/  2 CC39 CHAR(03) INIT('[K]'),  2 SOCIAL\_ID CHAR(16) INIT(' '), /\* ID ΠΕΛΑΤΗ ΣΙΤΙΣΗΣ \*/  /\* PAN MASKING FOR VISA \*/  2 CC40 CHAR(03) INIT('[L]'),  2 KEPID CHAR(10) INIT(' '), /\* ID KEP \*/  /\* APPLICATION DATE FOR \*/  /\* VISA \*/  2 CC41 CHAR(03) INIT('[M]'),  2 KEPNAME CHAR(27) INIT(' '), /\* NAME KEP \*/  /\* PRODUCT TEXT FOR VISA\*/  2 CC42 CHAR(03) INIT('[N]'),  2 PUR\_INT\_LIM PIC'ZZ.ZZ9V,99', /\* INT LIMIT 0 SITISH \*/  2 CC45 CHAR(03) INIT('[O]'),  2 CARD\_KIND CHAR(01) INIT(' '), /\*1:PLASTIC,2:MT,3:BOTH \*/  2 MT\_KIND CHAR(01) INIT(' '), /\*0:NONE,1:BRACELET,... \*/  2 MT\_COLOR CHAR(01) INIT(' '), /\*0:NONE,1:DEFAULT,.... \*/  2 FIL\_09 CHAR(05) INIT(' '),  /\* USER DATA \*/  /\* CHIP DATA \*/  2 CHIP\_CVC PIC'999', /\*USER DATA\*/  2 PIN\_BLOCK CHAR(16) INIT(' '), /\*FORMAT 5 \*/  2 FILL\_10 CHAR(01) INIT(' '),  2 DELIV\_TYPE CHAR(01) INIT(' '), /\* 8 FOR MASS ΣΙΤΙΣΗ \*/  2 FILL\_11 CHAR(03) INIT(' '),  /\* MARK END OF CARD \*/  2 CC43 CHAR(05) INIT('{END{'),  2 CC44 CHAR(06) INIT('@@@@@@'); |
| **OUTFILE2 FILE DECLARATION** | **KS90POU2 : EMBOSSING FILE FOR NEXT PGM (ALL)**  DCL 1 REMBOALL,  2 KINDCNT PIC '(06)9' INIT(0),  2 HOLDER CHAR (026) INIT(' '),  2 COMPANY CHAR (026) INIT(' '),  2 TRACK1\_CARD CHAR (016) INIT(' '),  2 STREET CHAR (025) INIT(' '),  2 ZIPCODE CHAR (005) INIT(' '),  2 CITY CHAR (010) INIT(' '),  2 FILE\_EMB\_NAM CHAR (003) INIT(' '),  2 FILE\_EMB\_CNT PIC '99999' INIT(' '),  2 PLS\_DEL\_TYPE CHAR (001) INIT(' '); |
| **OUTFILE3 FILE DECLARATION** | **KS90POU4 : EMBOSSING FILE FOR SSIS (MC)**  DCL 1 REMBOSPM,  /\* 1. EMBOSSING AREA (Emb) \*/  2 KINDCNT PIC '(06)9' INIT(0),  2 CC01 CHAR (001) INIT('{'),  2 CC02 CHAR (003) INIT('DCC'),  2 CC03 CHAR (001) INIT('{'),  2 CC04 CHAR (001) INIT('{'),  2 CC05 CHAR (003) INIT('EMB'),  2 CC06 CHAR (001) INIT('{'),  2 LINE1\_CARD CHAR (019) INIT(' '),  2 CC07 CHAR (001) INIT('}'),  2 FILLER01 CHAR (012) INIT(' '),  2 MM\_LHX CHAR (002) INIT(' '),  2 CC08 CHAR (001) INIT('/'),  2 YY\_LHX CHAR (002) INIT(' '),  2 FILLER02 CHAR (010) INIT(' '),  2 CC09 CHAR (001) INIT('}'),  2 HOLDER CHAR (026) INIT(' '),  2 CC10 CHAR (001) INIT('}'),  2 COMPANY CHAR (026) INIT(' '),  /\* 2. INDENT AREA (Ind) \*/  2 CC11 CHAR (001) INIT('{'),  2 CC12 CHAR (003) INIT('OCR'),  2 CC13 CHAR (001) INIT('{'),  2 FILLER03 CHAR (012) INIT(' '),  2 CARD\_4LAST PIC '9999' ,  2 FILLER04 CHAR (001) INIT(' '),  2 CVV2\_INDENT PIC '999' ,  2 FILLER05 CHAR (001) INIT(' '),  /\* 3. TRACK - 1 AREA (Tr1) \*/  2 CC14 CHAR (001) INIT('{'),  2 CC15 CHAR (003) INIT('ENC'),  2 CC16 CHAR (003) INIT('{%B'),  2 TRACK1\_CARD CHAR (016) INIT(' '),  2 CC17 CHAR (001) INIT('^'),  2 TRACK1\_NAME CHAR (026) ,  2 CC18 CHAR (001) INIT('^'),  2 YYMM\_LHXH\_1 PIC '9999' ,  2 SVC\_1 CHAR (003) INIT(' '),  2 CVV1\_1 PIC '999' ,  2 ZEROES\_10 CHAR (010) INIT('0000000000'),  2 CC19 CHAR (001) INIT('?'),  2 FILLER06 CHAR (001) INIT(' '),  /\* 4. TRACK - 2 AREA (Tr2) \*/  2 CC20 CHAR (001) INIT(';'),  2 TRACK2\_CARD CHAR (016) INIT(' '),  2 CC21 CHAR (001) INIT('='),  2 YYMM\_LHXH\_2 PIC '9999' ,  2 SVC\_2 CHAR (003) INIT(' '),  2 CVV1\_2 PIC '999' ,  2 CARD\_SEQ\_NUM PIC '99' ,  2 ZEROES\_8 CHAR (008) INIT('00000000'),  2 CC22 CHAR (001) INIT('?'),  /\* 5. INSERTER AREA (Ins) \*/  2 FILLER07 CHAR (001) INIT(' '),  2 CC23 CHAR (002) INIT('|1'),  2 CC24 CHAR (002) INIT('\_0'),  2 CC25 CHAR (011) INIT('[4037 0883]'),  2 TEMPL\_NAME CHAR (012) INIT(' '),  2 CC26 CHAR (003) INIT('[1]'),  2 KINDCNT2 CHAR (006) ,  2 CC27 CHAR (003) INIT('[2]'),  2 BRANCH CHAR (003) ,  2 CC28 CHAR (003) INIT('[3]'),  2 SNAME CHAR (022) ,  2 CC29 CHAR (003) INIT('[4]'),  2 PNAME CHAR (001) ,  2 CC30 CHAR (003) INIT('[5]'),  2 FNAME CHAR (003) ,  2 CC31 CHAR (003) INIT('[6]'),  2 STREET CHAR (025) ,  2 CC32 CHAR (003) INIT('[7]'),  2 ZIPCODE CHAR (005) ,  2 CC33 CHAR (003) INIT('[8]'),  2 CITY CHAR (010) ,  2 CC34 CHAR (003) INIT('[9]'),  2 HMER CHAR (010) INIT(' '),  2 CC35 CHAR (003) INIT('[A]'),  2 POS\_LIMIT /\* ? \*/ CHAR (009) ,  2 CC36 CHAR (003) INIT('[B]'),  2 WDRW\_LIMIT /\* ? \*/ CHAR (009) ,  2 CC37 CHAR (003) INIT('[C]'),  2 FILE\_EMB\_NAM /\* ? \*/ CHAR (003) INIT(' '),  2 FILE\_EMB\_CNT /\* ? \*/ PIC '99999' ,  2 CC38 CHAR (003) INIT('[D]'),  2 MATRIX /\* ? \*/ CHAR (016) INIT(' '),  2 CC39 CHAR (003) INIT('[K]'),  2 EMPLOYEE\_ID CHAR (016) INIT(' '),  2 CC40 CHAR (003) INIT('[L]'),  2 DEPARTMENT\_ID CHAR (010) INIT(' '),  2 CC41 CHAR (003) INIT('[M]'),  2 COMPANY\_NAME CHAR (027) INIT(' '),  2 CC42 CHAR (003) INIT('[N]'),  2 INT\_LIMIT /\* ? \*/ CHAR (009) ,  2 CC43 /\* ? \*/ CHAR (003) INIT(' '),  2 FILLER09 /\* ? \*/ CHAR (001) INIT(' '),  2 FILLER10 /\* ? \*/ CHAR (007) INIT(' '),  /\* 6. USER DATA (UsDa) \*/  2 ICVV\_USER PIC '999' ,  2 PIN\_BLOCK CHAR (016) INIT(' '),  2 FILLER11 CHAR (001) INIT(' '),  2 PLS\_DEL\_TYPE CHAR (001) INIT(' '),  2 DEVICE CHAR (001) INIT(' '),  2 PL\_COLOR CHAR (001) INIT(' '),  2 MT\_COLOR CHAR (001) INIT(' '),  2 CC44 CHAR (005) INIT('{END{'),  2 CC45 CHAR (006) INIT('@@@@@@'); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | @UCDEFD3 UPDATE CRDT.EMBOS\_FILE\_MASTER/CRDT.EMBOS\_FILE\_DETAIL  @SCDEPM, SELECT FROM CRDT.EMBOS\_PARM  @SCDECR, SELECT FROM CRDT.EMBOS\_COUNTER  @UCDECR, UPDATE FROM CRDT.EMBOS\_COUNTER  @SCCSPN1, SELECT FROM KCCT.SMARTPIN  @UCCSPN1, UPDATE KCCT.SMARTPIN  @UCDEFM2, UPDATE CRDT.EMBOS\_FILE\_MASTER  @SRPCRGM |
| **Main Process Logic** | * RETRIEVE CURRENT DATE * OPEN ALL OUTPUT FILES KS90POU1 FOR VISA , KS90POU2 FOR ALL CARDS , KS90POU3 FOR AS400 , KS90POU4 FOR MC , CMODFILE * Open cursor FROM CRDT.EMBOS\_FILE\_DETAIL A, CRDT.EMBOS\_FILE\_MASTER B With A.FILE\_ID = B.FILE\_ID AND A.KIND IN('NEW','PLA','MIG', 'RWL') AND A.MATURITY\_DATE <= :WS\_DATE\_CUR AND A.CONDITION = 5 AND CLASS\_PRODUCT = 'PC' ORDERED BY B.PRODUCTION\_CODE, B.PRODUCT\_CODE, B.WAY4\_FILE\_NAME, A.KIND * Fetch First Row and do the looping of next Tab * Depending if counter of previous step LYKOS\_STATUS\_CNT > 0 then UPDATE CRDT.EMBOS\_FILE\_MASTER by invoking service @UCDEFM2 * Close cursor , OUTPUT FILES KS90POU1 , KS90POU2 , KS90POU3 , KS90POU4 , CMODFILE * Print statistics counters to Console |
| **PROGRAM MAIN LOOP** | * For each line of DCLEMBOS cursor EXCLUDE PLASTIC REPLACEMENTS REQUESTS meaning DCLEMBOS\_FILE\_DETAIL.KIND different than 'RWL' , 'PLA' and DCLEMBOS\_FILE\_DETAIL.SMARTPIN = 'Y' and do the following   + RETRIEVE card details Data FROM KCCT.SMARTPIN using the service @SCCSPN1   + UPDATE KCCT.SMARTPIN WITH MASKED MATRIX invoking the service @UCCSPN1   + UPDATE KCCT.SMARTPIN FLG TO '1' except the case that FLG\_DELIVER = '3', FLG = '4' in order TO SEND SMS WITH PIN POSITIONS * We COMPARE PREVIOUS WAY4 FILE NAME WITH CURRENT (PREV\_WAY4\_FILE\_NAME ^= DCLEMBOS\_FILE\_MASTER.WAY4\_FILE\_NAME) and if there is difference we do   + IF (LYKOS\_STATUS\_CNT > 0 UPDATE CRDT.EMBOS\_FILE\_MASTER by invoking service @UCDEFM2   + INITIALIZE WORKING VARIABLES CONCERNING LYKOS COUNTS   + READ COUNTER FROM CRDT.EMBOS\_COUNTER using service @SCDECR   + UPDATE LAST COUNTER USED IN CRDT.EMBOS\_COUNTER using service @UCDECR * Increase counter LYKOS\_STATUS\_CNT * READ FILE NAME FROM CRDT.EMBOS\_PARM by invoking service @SCDEPM * PROCESS ITERATION DATA AND WRITE EITHER TO EMBOSSING FILE OR TO CMOD FILE depending on the field DCLEMBOS\_FILE\_DETAIL.SCHEME   + When MC mastercard , INITIALIZE PREPAID OUTPUT , and depending if product is instant or not (DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT ^= 'INS') we form different MC OUTPUT DECLARATION through different routines FORMAT\_OUTPUT\_MC, FORMAT\_INSOUT\_MC   + When Visa INITIALIZE OUTPUT VARIABLES and and depending if product is instant or not (DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT ^= 'INS') we form different OUTPUT DATA FORMAT\_OUTPUT\_VISA , FORMAT\_INSOUT\_VISA   + PERFORM VALIDATIONS for Data integrity   + WRITE TO OUTPUT when no errors in validations     - EVALUATE RECORD ID     - WRITE TO EMBOSSING FILE SENT TO LYKOS. When MC KS90POU4 , when VISA KS90POU1     - WRITE TO OUTPUT FILE 2 FOR FURTHER PROCESS by filling the fields dependiing Visa / MC and write to KS90POU2 * UPDATE CRDT.EMBOS\_FILE\_DETAIL WITH LYKOS FILE invoking service @UCDEFD3 * HOLD PREVIOUSLY FILE NAME READ IN TO LOCAL VARIABLE DCLEMBOS\_FILE\_MASTER.WAY4\_FILE\_NAME * FETCH NEXT RECORD |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER CRDT.EMBOS\_COUNTER CRDT.EMBOS\_PARM |

#### [KRPPD025] Embossing of prepaid products in WAY4 (Phase 3)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRPPD025** | KS96P | CRDP.PD.PRKEA.INTT.CRD02409.SEQ | Ν/Α | Process file which will be sent to SSIS and then to LYKOS.  All records which will be sent to LYKOS should be logged to KCCT.TRACK\_INFO table (before transfer to SSIS) |
|  | SORT | CRDP.PD.PRKEA.INTT.CRD02409.SEQ | CRDP.PD.PRKEA.SEMT.CRD02501.SEQ | Process embossing file and add 1 extra character at the end of file |
|  | KS91P | CRDP.PD.PRKEA.SEMT.CRD02501.SEQ | N/A | Update table CRDT.EMBOS\_FILE\_DETAIL and set condition from 5 to 6 |
|  | KS92P | N/A | CRDP.PD.PRKEA.CMOD.CRD02501.SEQ | Produce CMOD file for all records which exist in CRDT.EMBOS\_FILE\_DEATIL table with condition 8 or 9 |
|  | KS93P | N/A | CRDP.PD.PRKEA.CMOD.CRD02502.SEQ | Create a CMOD statistics file regarding LYKOS daily file |
|  | KS97P | N/A | CRDP.PD.PRKEA.SEMT.CRD02502.SEQ,  CRDP.PD.KANON.FTOP.CRD02502.SEQ | Collect all rejected cards from embossing outgoing process and sends the response to SSIS in order for the response to be sent to WAY4 |
|  | SORT | CRDP.PD.PRKEA.SEMT.CRD02502.SEQ | CRDP.PD.PRKEA.FTOP.CRD02500.SEQ | Rejected cards:  Insert header, trailer, and identity for rows (FH,FD,FT) |
|  | KS95P | CRDP.PD.PRKEA.INTT.CRD02409.SEQ | CRDP.PD.PRKEA.SEMT.CRD02503.SEQ | Read outgoing file that will be sent to LYKOS and produce a file with unique mailer code in order to update WAY4 |
|  | SORT | CRDP.PD.PRKEA.SEMT.CRD02503.SEQ | CRDP.PD.PRKEA.FTOP.CRD02501.SEQ | Mailer:  Insert header, trailer, and identity for rows (FH,FD,FT) |
|  | IDCAMS | CRDP.PD.PRKEA.FTOP.CRD02501.SEQ | KATP.PD.PRKEA.GDGT.CRD02501.BACK(+1) | Create meiler file backup |
|  | IDCAMS | CRDP.PD.PRKEA.FTOP.CRD02500.SEQ | KATP.PD.PRKEA.GDGT.CRD02500.BACK(+1) | Create rejected file backup |
|  | IRXJCL | N/A | N/A | Report unfulfilled requests issuing/reissuing (CMOD):  Run RUNCMOD CRDP.PD.PRKEA.CMOD.CRD02501.SEQ CRD023DPL CDLPL |
|  | IRXJCL | N/A | N/A | Report unfulfilled requests issuing/reissuing VISA PC KA cards (CMOD):  Run: RUNCMOD CRDP.PD.PRKEA.CMOD.CRD02502.SEQ CRD024DPL CDLPL |
|  | IEFBR14 | CRDP.PD.PRKEA.INTT.CRD02409.SEQ,  CRDP.PD.PRKEA.SEMT.CRD02501.SEQ,  CRDP.PD.PRKEA.SEMT.CRD02502.SEQ,  CRDP.PD.PRKEA.SEMT.CRD02503.SEQ | N/A | Deallocation of working files |

#### 

#### [KS96P] Process file for SSIS and LYKOS, Log records to table

| **IL Process** | |
| --- | --- |
|  | |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | KS96PI: Embossing file containing all records that will be sent to LYKOS  DCL 1 REMBOS,  2 KINDCNT PIC '(06)9' INIT(0),  2 HOLDER CHAR (026) INIT(' '),  2 COMPANY CHAR (026) INIT(' '),  2 TRACK1\_CARD CHAR (016) INIT(' '),  2 STREET CHAR (025) INIT(' '),  2 ZIPCODE CHAR (005) INIT(' '),  2 CITY CHAR (010) INIT(' '),  2 FILE\_EMB\_NAM CHAR (003) INIT(' '),  2 FILE\_EMB\_CNT PIC '99999' INIT(' '),  2 PLS\_DEL\_TYPE CHAR (001) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | READ\_KS96PI: More lines in input file |
| **SERVICES** | ICDTRC: Insert into KCCT.TRACK\_INFO  SCDEFD2: Select from CRDT.EMBOS\_FILE\_DETAIL  SCDTRC: Select latest record from KCCT.TRACK\_INFO for card |
| **Main Process Logic** | Retrieve current date.  For every line in input file: Retrieve information from tables. Insert details into table, and commit every 20 inserts |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every record of input file:   * Retrieve details from CRDT.EMBOS\_FILE\_DETAIL (by calling SCDEFD2) * Retrieve latest track of card (by calling SCDTRC) * Insert details into KCCT.TRACK\_INFO (by calling ICDTRC), and commit every 20 inserts |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | KCCT.TRACK\_INFO  CRDT.EMBOS\_FILE\_DETAIL |

#### [KS91P] Update table and set condition form 5 to 6

| **IL Process** | |
| --- | --- |
|  | |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | KS91PIN:  DCL 1 RKS91PIN,  2 KINDCNT PIC '(06)9' INIT(0),  2 HOLDER CHAR (026) INIT(' '),  2 COMPANY CHAR (026) INIT(' '),  2 TRACK1\_CARD CHAR (016) INIT(' '),  2 STREET CHAR (025) INIT(' '),  2 ZIPCODE CHAR (005) INIT(' '),  2 CITY CHAR (010) INIT(' '),  2 FILE\_EMB\_NAM CHAR (003) INIT(' '),  2 FILE\_EMB\_CNT PIC '99999' INIT(' '),  2 PLS\_DEL\_TYPE CHAR (001) INIT(' '),  2 UPD\_FLG CHAR (001) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_KS91PIN: More lines in input file |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve the current date.  Check for incomplete requests, by selecting from CRDT.EMBOS\_FILE\_DETAIL with kind NEW/PLA/PIN/MIG/RWL, product code 143/144/145/146, smartpin enabled, and condition 4.  For every line in input file: Retrieve card details from table with condition 5. Update record status and condition from 5 to 6, and set flag to input file for said update. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every record of input file:   * Retrieve card details from CRDT.EMBOS\_FILE\_DETAIL with condition 5 * **If** card with condition 5 was found (DCLEMBOS\_FILE\_DETAIL.CONDITION = 5) **then**:   + Update record in table table UPDATE\_DETAILS\_RTN with record status and condition 6 (and update timestamp with current time), and execute commit   + Rewrite input file, to set update flag (RKS91PIN.UPD\_FLG = '1') * **Else if** card with condition lower/higher than 5 was found **then** Error accordingly (05/06) |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER  CRDT.SMARTPIN\_TRACK  KCCT.SMARTPIN |

#### [KS92P] Produce CMOD file for all records in table with conditions 8,9

| **IL Process** | |
| --- | --- |
|  | |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | KS92POUT:  DCL FLINE CHAR(133) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve the current date.  Retrieve details from CRDT.EMBOS\_FILE\_DETAIL for kind NEW/PLA/PIN/MIG/RWL, maturity date in the past 5 days, condition 8/9, and Product 143/144/145/146.  For every row in query: Update current branch local variable if needed. Write information to the output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every record of query:   * **If** saved branch is different than details’ issuing branch (WS\_BRANCH != SUBSTR(DCLEMBOS\_FILE\_DETAIL.ISSUING\_BRANCH,1,3)) **then**:   + Update local branch variable (WS\_BRANCH) and set page initialisation flag and serial number (WS\_CNTREAD\_BRANCH) for branch cards * Write data to output file:   + '1 '!!'ΕΘΝΙΚΗ ΤΡΑΠΕΖΑ ΤΗΣ ΕΛΛΑΔΟΣ Α.Ε'   !!(59)' '!!'ΚΩΔΙΚΟΣ ΚΑΤΑΣΤΑΣΗΣ: CRD023-DPL'   * + ' '!!'ΠΡΟΣ ΚΑΤΑΣΤΗΜΑ: '!!WS\_BRANCH!!' - '   !!WS\_BRANCH\_TITLE  !!(41)' '!!'ΗΜΕΡΑ: '!!WS\_DATE\_CUR  !!' '!!'(ΩΡΑ: '!!WS\_TIME\_CUR!!')'   * + ' '!!'ΚΑΤΑΣΤΑΣΗ '   !!'ΑΝΕΚΠΛΗΡΩΤΩΝ ΑΙΤΗΜΑΤΩΝ ΕΚΔΟΣΕΩΝ/'  !!'ΕΠΑΝΕΚΔΟΣΕΩΝ '  !!(54)' '!!'ΣΕΛΙΣ '!!ANALPAGE   * + ' '!!(03)' '!!'Α/Α'   !!(07)' '!!'ΚΑΡΤΑ'  !!(15)' '!!'ΟΝΟΜΑΤΕΠΩΝΥΜΟ'  !!(13)' '!!'ΣΥ.ΔΙ.ΠΕΛ.'  !!(06)' '!!'ΑΙΤΗΜΑ'  !!(08)' '!!'ΚΩΔΙΚΟΣ / AΙΤΙΟΛΟΓΙΑ ΑΠΟΡΡΙΨΗΣ'   * + FLINE=WS\_CNTREAD\_BRANCH!!(5)' ' !!   DCLEMBOS\_FILE\_DETAIL.CARD !!  FSTR\_CUSTNAME!!' ' !!  DCLEMBOS\_FILE\_DETAIL.I\_IP!!(01)' ' !!  WS\_CATEGORY!!(01)' ' !!  DCLEMBOS\_FILE\_DETAIL.REJECT\_REASON!!  (01)' '!!SUBSTR(REJECT\_REASON\_TXT,1,40);   * + Note: DCLEMBOS\_FILE\_DETAIL.KIND='NEW': WS\_CATEGORY=' ΝΕΑ ΕΚΔΟΣΗ '   DCLEMBOS\_FILE\_DETAIL.KIND='PIN': WS\_CATEGORY=' ΕΠΑΝΕΚΔΟΣΗ PIN '  DCLEMBOS\_FILE\_DETAIL.KIND='PLA': WS\_CATEGORY='ΕΠΑΝ/ΣΗ ΠΛΑΣΤΙΚΟΥ '  DCLEMBOS\_FILE\_DETAIL.KIND='MIG': WS\_CATEGORY=' MIGRATION '  DCLEMBOS\_FILE\_DETAIL.KIND='RWL': WS\_CATEGORY='ΕΠΑΝ/ΣΗ ΛΟΓΩ ΛΗΞΗΣ' |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### [KS93P] Create CMOD statistics file regarding LYKOS daily file

| **IL Process** | |
| --- | --- |
|  | |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | OUTFILE1: for CMOD  DCL ULINE CHAR (133) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve current date.  Retrieve details (count:L\_FCNT, issuing branch) from CRDT.EMBOS\_FILE\_DETAIL for kind MIG/NEW/PLA/RWL, update time in the past 3 hours, condition 6, and product code 143/144/145/146.  For every row in query: Retrieve details, and write data to output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every record of query:   * Init lines counter per branch change * Init page per branch change * Retrieve branch title from DXPT.GEOST * Retrieve details (product code, kind, count) per issuing branch from CRDT.EMBOS\_FILE\_DETAIL for kind MIG/NEW/PLA/RWL, update timestamp in the past 3 hours, condition 6, and product code 143/144/145/146   + Evaluate result of fetched count, based on local variable L\_KDPR = L\_FKND||L\_FPRD, and set value to relevant count to given count (other: insert count to L\_OTHCNT):   MIG144  NEW144  PLA144  RWL144  MAS144  MIG143  NEW143  PLA143  RWL143  MAS143  MIG146  NEW146  PLA146  RWL146  MAS146  MIG145  NEW145  PLA145  RWL145  MAS145   * Write data to output file for CMOD   + '1 '!!'ΕΘΝΙΚΗ ΤΡΑΠΕΖΑ ΤΗΣ ΕΛΛΑΔΟΣ Α.Ε.'   + ' '!!'ΚΩΔΙΚΟΣ ΚΑΤΑΣΤΑΣΗΣ : CRD024-DPL'   + ' '!!'ΠΡΟΣ KATAΣΤΗΜΑ: '!! WS\_BRANCH!!' - '!!WS\_BRANCH\_TITLE   + ' '!!'ΗΜΕΡΑ: '!!SUBSTR(L\_DATE\_CUR,1,2)!!'.'   !!SUBSTR(L\_DATE\_CUR,4,2)!!'.'!!SUBSTR(L\_DATE\_CUR,7,4)   * + (3)' '!!'ΚΑΤΑΣΤΑΣΗ ΑΙΤΗΜΑΤΩΝ ΕΚΔΟΣΕΩΝ/ΕΠΑΝΕΚΔΟΣΕΩΝ'   !!' VISA/MC PC ΕΤΕ / ΚΑ ΚΑΡΤΩΝ'!!(41)' '!!'ΣΕΛΙΣ '!!L\_PAGE;   * + ' '!!(131)'-'   + ' '!!'ΣΥΝΟΛΙΚΟΣ ΑΡΙΘΜΟΣ ΚΑΡΤΩΝ: '!!DEC\_TO\_CHAR(L\_FCNT)   + ' '!!(131)'-'   + ' '!!'ΑΝΑΛΥΤΙΚΑ: '   + ' '!!(131)'-'   + Specifically, the following products get written as follows:     - 'VISA PREPAID KA CARD',   '144',  L\_NEWCNT144,  L\_MIGCNT144,  L\_PLACNT144,  L\_RWLCNT144,  L\_MASCNT144   * + - 'VISA PREPAID ETE CARD',   '143',  L\_NEWCNT143,  L\_MIGCNT143,  L\_PLACNT143,  L\_RWLCNT143,  L\_MASCNT143   * + - 'MC PREPAID KA CARD',   '146',  L\_NEWCNT146,  L\_MIGCNT146,  L\_PLACNT146,  L\_RWLCNT146,  L\_MASCNT146   * + - 'MC PREPAID ETE CARD',   '145',  L\_NEWCNT145,  L\_MIGCNT145,  L\_PLACNT145,  L\_RWLCNT145,  L\_MASCNT145   * + ' '!!(131)'-'   + ' ΛΟΙΠΟΙ ΣΥΝΔΥΑΣΜΟΙ ΕΙΔΩΝ ΚΑΙ ΠΡΟΙΟΝΤΩΝ'   + ' '!!(131)'-'   + ' ΛΟΙΠΑ (ΣΥΝΟΛΙΚΑ) : '!!DEC\_TO\_CHAR(L\_OTHCNT)   + ' '!!(131)'-' |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### [KS97P] Collect rejected cards and respond to SSIS

| **IL Process** | |
| --- | --- |
|  | |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | KS97PO: Response file for SSIS for all rejected cards in outgoing embossing process  DCL 1 RKS97PO,  2 O\_REC\_NO CHAR (06) INIT(' '),  2 O\_CARD\_EMB CHAR (26) INIT(' '),  2 O\_CARD\_NO CHAR (16) INIT(' '),  2 O\_FILLER CHAR (02) INIT(' '),  2 O\_REJ\_CODE CHAR (03) INIT(' '),  2 O\_REJ\_DESCR CHAR (100) INIT(' '),  2 O\_FILE CHAR (15) INIT(' '),  2 O\_ELTA\_DATE CHAR (10) INIT(' '),  2 O\_ORIG\_REC\_ID CHAR (06) INIT(' '),  2 O\_EXCEPT\_FLG CHAR (01) INIT(' '),  2 O\_PLS\_DEL\_TYPE CHAR (02) INIT(' '),  2 WAY4\_FILE\_NAME CHAR (40) INIT(' '),  2 TRANSACTION\_ID CHAR (255) INIT(' '),  2 BATCH\_ID CHAR (32) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | KS97P2O:  DCL 1 RKS97P2O,  2 CARD CHAR (16) INIT(' '),  2 FILLER\_1 CHAR (01) INIT(';'),  2 SYDIPEL CHAR (11) INIT(' '),  2 FILLER\_2 CHAR (01) INIT(';'),  2 CLASSIFIER\_VALUE CHAR (01) INIT(' '),  2 FILLER\_3 CHAR (01) INIT(';'),  2 CLASSIFIER\_NAME CHAR (16) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | UCDEFD2: Update into CRDT.EMBOS\_FILE\_DETAIL  SRVIXAS: Retrieve customer SYDIPEL from card number |
| **Main Process Logic** | Retrieve current date.  Retrieve details from CRDT.EMBOS\_FILE\_DETAIL/CRDT.EMBOS\_FILE\_MASTER with record status 0,9,10, condition 9/10/13/14, product code 143/144/145/146.  For every row in query: Write data to output file 1, considering rejected conditions. For specified cases, write data to output file 2. Update record details in table CRDT.EMBOS\_FILE\_DETAIL. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every record of query:   * Write data to output file 1   + For condition 9 (DCLEMBOS\_FILE\_DETAIL.CONDITION = 9), reject:     - Reject code (RKS97PO.O\_REJ\_CODE) ‘009’ and description (RKS97PO.O\_REJ\_DESCR) ‘CANCELATION’   + For condition 13, reject:     - Reject code ‘000’ and description ‘REGARDING FLOW’   + For condition 10, reject:     - Reject code ‘010’ and description ‘VALIDATION FAIL’ * For condition 14 (DCLEMBOS\_FILE\_DETAIL.CONDITION = 14) and reject reason not C5 (DCLEMBOS\_FILE\_DETAIL.REJECT\_REASON <> 'C5')   Or production event RCRD (DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT= 'RCRD')   * + Write data to output file 2 (classifier name CLS\_CARD\_RENEWAL) * Update record in CRDT.EMBOS\_FILE\_DETAIL to record status/condition 11 (by calling UCDEFD2) |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | KCCT.TRACK\_INFO  KCCT.CARDS\_PIN  CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER |

#### [KS95P] Read outgoing file (for LYKOS), Produce file with unique Mailer Code (for WAY4)

| **IL Process** | |
| --- | --- |
|  | |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | KS95PI:  DCL 1 I\_REC,  2 KINDCNT PIC '(06)9' INIT(0),  2 HOLDER CHAR (026) INIT(' '),  2 COMPANY CHAR (026) INIT(' '),  2 TRACK1\_CARD CHAR (016) INIT(' '),  2 STREET CHAR (025) INIT(' '),  2 ZIPCODE CHAR (005) INIT(' '),  2 CITY CHAR (010) INIT(' '),  2 FILE\_EMB\_NAM CHAR (003) INIT(' '),  2 FILE\_EMB\_CNT PIC '99999' INIT(' '),  2 PLS\_DEL\_TYPE CHAR (001) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | KS95PO:  DCL 1 RKS95PO,  2 CARD\_NO CHAR(16) INIT(' '),  2 FILE CHAR(15) INIT(' '),  2 ORIG\_REC\_ID CHAR(06) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | READ\_KS95PI: More lines in input file |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve current date.  For every row in input file: Write data to output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every record of input file:   * Write data to output file:   + Note, RKS95PO.FILE = I\_REC.FILE\_EMB\_NAM!!I\_REC.FILE\_EMB\_CNT and RKS95PO.ORIG\_REC\_ID = I\_REC.KINDCNT |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

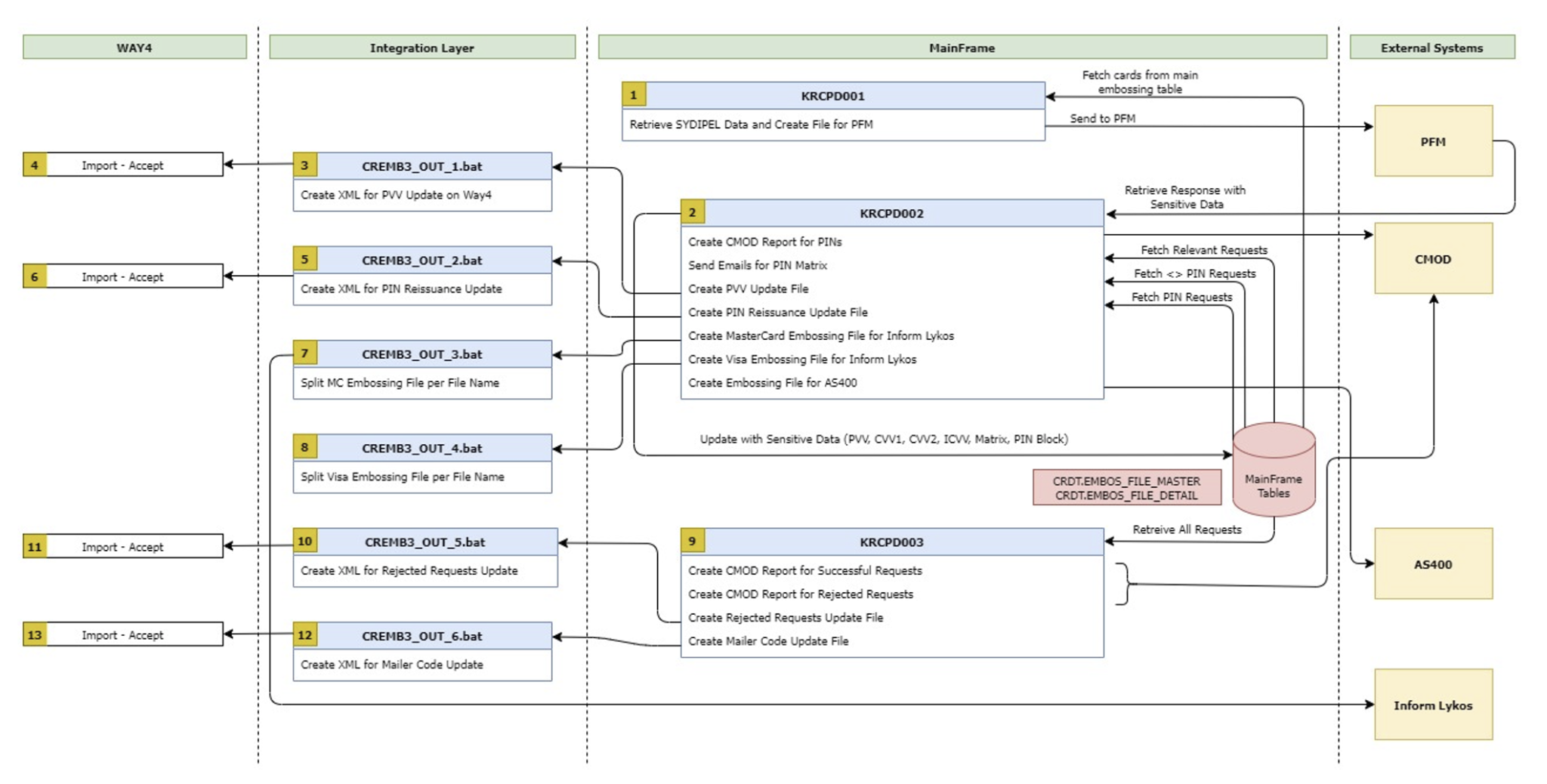
## 

## [Embossing - Credit]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Massive Loading Voucher |
| **Batch Flow Title** | KRPPD013-015 & KRPPD017-019 |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | Way4 |
| **Target Systems** | Mainframe z/OS |
| SRVCDAC**Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks:   The credit embossing process uses client and cards tables to collect and validate the appropriate data that are required for the process. It sends a file to PFM for the cryptographic card data to be produced. It receives a file from PFM and delivers the embossing file to IL in order to be sent to Inform Lykos. |



### 

### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KRCPD001] Enrich table with customer data and create file for PFM

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRCPD001** | K03DP | N/A | N/A | Retrieve cardholder’s details from SYDIPEL |
|  | K04DP | N/A | CRDP.PD.CCEMB.SEMT.CRD00101.SEQ,  CRDP.PD.CCEMB.SEMT.CRD00102.SEQ | Create file for PFM |
|  | IDCAMS | CRDP.PD.CCEMB.SEMT.CRD00101.SEQ | KATP.PD.CCEMB.GDGT.CRD00101.BACK(+1) | Backup the FTOP files to PFM |
|  | IDCAMS | CRDP.PD.CCEMB.SEMT.CRD00101.SEQ | CRDP.PD.CCEMB.FTOP.CRD00101.SEQ | Copy PFM file without header |
|  | K05DP | CRDP.PD.CCEMB.SEMT.CRD00102.SEQ | N/A | Update CRDT.EMBOS\_FILE\_DETAIL with CONDITION=1 and insert row to KCCT.TRACK\_INFO |
|  | K06DP | CRDP.PD.CCEMB.SEMT.CRD00102.SEQ | N/A | Create CMOD file for new issuanses and plastic replacements |
|  | SORT | CRDP.PD.CCEMB.SEMT.CRD00101.SEQ | CRDP.PD.CCEMB.INTT.CRD00104.SEQ | Create file to keep PFM file header for later use |
|  | IEFBR14 | CRDP.PD.CCEMB.SEMT.CRD00101.SEQ,  CRDP.PD.CCEMB.SEMT.CRD00102.SEQ |  | Deallocation of working files |

#### 

#### [KRCPD002] Add sensitive data from PFM to table & send PIN reminders (Emails/SMS)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRCPD002** | IDCAMS | CRDP.PD.CCEMB.INTT.CRD00104.SEQ,  CRDP.PD.CCEMB.FTIP.CREDCPFM.SEQ | CRDP.PD.CCEMB.SEMT.CRD00201.SEQ | Copy the file with the header and the PFM file to a new file |
|  | IDCAMS | CRDP.PD.CCEMB.SEMT.CRD00201.SEQ | KATP.PD.CCEMB.GDGT.CRDPFM24.BACK(+1) | Create backup for file form PFM |
|  | K07DP | CRDP.PD.CCEMB.SEMT.CRD00201.SEQ | N/A | Update table EMBOS\_FILE\_DETAIL and set condition from 2 to 3 |
|  | K08DP | CRDP.PD.CCEMB.FTIP.CREDCPFM.SEQ | N/A | Update table EMBOS\_FILE\_DETAIL and set condition from 3 to 4 |
|  | K09DP | N/A | N/A | Check not processed requests |
|  | K10DP | CRDP.PD.CCEMB.PERP.CRD00201.SEQ | N/A | Create ‘PAPYROUS’ file for the data we have up to now |
|  | K11DP | N/A | N/A | Process all requests for smartpin reissuance, send email for smartpin for reissued credit cards |
|  | K12DP | N/A | CRDP.PD.CCEMB.SEMT.CRD00202.SEQ | Select PLA/RWL/NEW/MIG requests from CRDT.EMBOS\_FILE\_DETAIL, update conditions from 4 to 5, write to file only those with NEW/MAS/MIG for the update of PVV of those cards in WAY4 |
|  | SORT | CRDP.PD.CCEMB.SEMT.CRD00202.SEQ | CRDP.PD.CCEMB.FTOP.CRD00200.SEQ | PVV update: Insert Header, Trailer, Identity for rows (FH,FD,FT) |
|  | K13DP | N/A | CRDP.PD.CCEMB.SEMT.CRD00203.SEQ | Create file with all PIN reissuance requests |
|  | SORT | CRDP.PD.CCEMB.SEMT.CRD00203.SEQ | CRDP.PD.CCEMB.FTOP.CRD00201.SEQ | PIN reissuance: Insert HEADER, Trailer, Identity for rows (FH,FD,FT) |
|  | K14DP | N/A | CRDP.PD.CCEMB.SEMT.CRD00204.SEQ,  CRDP.PD.CCEMB.SEMT.CRD00205.SEQ,  CRDP.PD.CCEMB.INTT.CRD00206.SEQ,  CRDP.PD.CCEMB.INTT.CRD00207.SEQ | Create Embossing file to send to LYKOS |
|  | SORT | CRDP.PD.CCEMB.SEMT.CRD00204.SEQ | CRDP.PD.CCEMB.FTOP.CRD00202.SEQ | Embossing file (Mastercard): Insert HEADER, Trailer, Identity for rows (FH,FD,FT) |
|  | SORT | CRDP.PD.CCEMB.SEMT.CRD00205.SEQ | CRDP.PD.CCEMB.FTOP.CRD00203.SEQ | Embossing file (Visa): Insert HEADER, Trailer, Identity for rows (FH,FD,FT) |
|  | IDCAMS | CRDP.PD.CCEMB.FTOP.CRD00200.SEQ | KATP.PD.CCEMB.GDGT.CRD00200.BACK(+1) | Create PVV update backup file |
|  | IDCAMS | CRDP.PD.CCEMB.FTOP.CRD00201.SEQ | KATP.PD.CCEMB.GDGT.CRD00201.BACK(+1) | Create PIN reissuance backup file |
|  | IDCAMS | CRDP.PD.CCEMB.FTOP.CRD00202.SEQ | KATP.PD.CCEMB.GDGT.CRD00202.BACK(+1) | Create Embossing file backup (Mastercard) |
|  | IDCAMS | CRDP.PD.CCEMB.FTOP.CRD00203.SEQ | KATP.PD.CCEMB.GDGT.CRD00203.BACK(+1) | Create Embossing file backup (VIsa) |
|  | IEFBR14 | CRDP.PD.CCEMB.SEMT.CRD00201.SEQ,  CRDP.PD.CCEMB.SEMT.CRD00202.SEQ,  CRDP.PD.CCEMB.SEMT.CRD00203.SEQ,  CRDP.PD.CCEMB.SEMT.CRD00204.SEQ,  CRDP.PD.CCEMB.SEMT.CRD00205.SEQ,  CRDP.PD.CCEMB.INTT.CRD00104.SEQ | N/A | Deallocation of working files |
|  | CLRSEQ | CRDP.PD.CCEMB.FTIP.CREDCPFM.SEQ | N/A | Clear file retrieved from PFM (without header) |

#### 

#### [KRCPD003] Insert and update card tables, create files for WAY4/LYKOS

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRCPD003** | K15DP | CRDP.PD.CCEMB.INTT.CRD00206.SEQ | N/A | Process file which will be sent to SSIS and then to LYKOS.  Log relevant records into KCCT.TRACK\_INFO (before transfer to SSIS) |
|  | K17DP | CRDP.PD.CCEMB.INTT.CRD00206.SEQ | N/A | Update table EMBOS\_FILE\_DETAIL and set condition from 5 to 6 |
|  | K18DP | N/A | N/A | **?????????????????????????????**  **?????????????????????????????** |
|  | K19DP | N/A | N/A | **?????????????????????????????**  **?????????????????????????????** |
|  | K20DP | N/A | CRDP.PD.CCEMB.SEMT.CRD00301.SEQ,  CRDP.PD.KANON.FTOP.CRD00302.SEQ | Collect all rejected cards from embossing outgoing process and send the response to SSIS in order for the response to be sent to WAY4 |
|  | SORT | CRDP.PD.CCEMB.SEMT.CRD00301.SEQ | CRDP.PD.CCEMB.FTOP.CRD00300.SEQ | Rejected cards: Insert HEADER, Trailer, Identity for rows (FH,FD,FT) |
|  | K21DP | CRDP.PD.CCEMB.INTT.CRD00206.SEQ | CRDP.PD.CCEMB.SEMT.CRD00302.SEQ | Read outgoing file that will be sent to LYKOS and produce a file with unique mailer code in order to update WAY4 |
|  | SORT | CRDP.PD.CCEMB.SEMT.CRD00302.SEQ | CRDP.PD.CCEMB.FTOP.CRD00301.SEQ | Mailer: Insert HEADER, Trailer, Identity for rows (FH,FD,FT) |
|  | IDCAMS | CRDP.PD.CCEMB.FTOP.CRD00301.SEQ | KATP.PD.CCEMB.GDGT.CRD00301.BACK(+1) | Create mailer file backup |
|  | IDCAMS | CRDP.PD.CCEMB.FTOP.CRD00300.SEQ | KATP.PD.CCEMB.GDGT.CRD00300.BACK(+1) | Create rejected file backup |
|  | IEFBR14 | CRDP.PD.CCEMB.INTT.CRD00206.SEQ,  CRDP.PD.CCEMB.SEMT.CRD00301.SEQ,  CRDP.PD.CCEMB.SEMT.CRD00302.SEQ | N/A | Deallocation of working files |

#### 

#### [K03DP] Retrieve cardholder’s details from SYDIPEL

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | SRVIXAS: Retrieve card holder’s SYDIPEL (active relation)  SRVIIPS: Retrieve cats holder’s basic information (using SYDIPEL)  SRVILOC: Retrieve classification address  SRVIADR: Retrieve address  SRVINMF: Retrieve names  SRVINDS: Retrieve customer’s SYDIPEL indicators |
| **Main Process Logic** | Retrieve current date.  For every record returned from query to CRDT.EMBOS\_FILE\_DETAIL: Check request compliance with card status according to the specific record’s case. If status is found to agree then for production event Instant set personal type, otherwise retrieve customer data. If status is found to agree, update table. |
| **PROGRAM MAIN LOOP** | For every record returned from query to CRDT.EMBOS\_FILE\_DETAIL:   * Check request with card status:   + For kind NEW, Migration (MIG) or PLA that are either active status (DCLEMBOS\_FILE\_DETAIL.CARD\_STATUS != '0'), PIN production status not 2 (DCLEMBOS\_FILE\_DETAIL.PIN\_PROD\_STATUS != '2'), Plastic production status not S (DCLEMBOS\_FILE\_DETAIL.PLA\_PROD\_STATUS != S), or Plastic status not I (DCLEMBOS\_FILE\_DETAIL.PLA\_STATUS != I):     - Set FOUND\_OK flag to 1 (they don’t agree)     - Update table CRDT.EMBOS\_FILE\_DETAIL   + For kind PIN with either active status (DCLEMBOS\_FILE\_DETAIL.CARD\_STATUS != '0'), PIN production status not 2 (DCLEMBOS\_FILE\_DETAIL.PIN\_PROD\_STATUS != '2'):     - Set FOUND\_OK flag to 1 (they don’t agree)     - Update table CRDT.EMBOS\_FILE\_DETAIL     - Remove SMARTPIN request for rejected card by updating KCCT.SMARTPIN/KCCT.SMARTPIN\_HIST and deleting related record from KCCT.SMARTPIN   + For kind RWL (Renewal) with either active status (DCLEMBOS\_FILE\_DETAIL.CARD\_STATUS != '0'), Plastic production status not S (DCLEMBOS\_FILE\_DETAIL.PLA\_PROD\_STATUS != S), or Plastic status not I (DCLEMBOS\_FILE\_DETAIL.PLA\_STATUS != I):     - Set FOUND\_OK flag to 1 (they don’t agree)     - Update table CRDT.EMBOS\_FILE\_DETAIL * **If** status is found to agree (FOUND\_OK=0), **then**:   + **If** Production Event is Instant (DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT = INS) **then**:     - For product group value 06, set PERSONAL\_TYPE to Legal Entity (ΝΠ) otherwise set it to Individual (ΦΠ), and set FOUND\_PERSON\_TYPE flag to 1   + Else, get customer data:     - Find customer SYDIPEL by card number (by calling SRVIXAS)     - For Individual customer (@SRVIXAS.ODATA.CIP.C\_IP\_TP = 'ΦΠ'), look for ιindividual with sole proprietorship (@SRVIIPS.ODATA.I\_CLSF\_EMP\_ST\_K = 1200002) (by calling SRVIIPS) and if so, set FOUND\_FP\_DEPOSIT flag.     - **If** SYDIPEL was found (FOUND\_I\_IP=1) **then**:       * Retrieve names (by calling SRVINMF) and set FOUND\_NAMES to 1       * Retrieve classification address (by calling SRVILOC,SRVIADR),and set FOUND\_ADDRESS flag to 1     - For kind NEW/PLA/MIG(Migration)/RWL(Renewal), **if** SYDIPEL was found (FOUND\_I\_IP=1) and person type (FOUND\_PERSON\_TYPE=1) and address type (FOUND\_ADDRESS=1) and names (FOUND\_NAMES=1) **then** set FOUND\_OK flag to 0 and keep on     - For kind PIN, **if**  person type (FOUND\_PERSON\_TYPE=1) and address type (FOUND\_ADDRESS=1) and names (FOUND\_NAMES=1) **then** set FOUND\_OK flag to 0 and keep on     - Otherwise, Error     - For kind RWL (Renewal), check (by calling SRVINDS)       * if customer is blocked (@SRVINDS.ODATA.I\_CLSF\_INDC\_VAL\_K = 8900022)       * if customer is under investigation (@SRVINDS.ODATA.I\_CLSF\_INDC\_VAL\_K = 8900024)       * if transaction is prohibited for customer (@SRVINDS.ODATA.I\_CLSF\_INDC\_VAL\_K = 8300001)       * If customer has passed away (@SRVIIPS.ODATA.I\_CLSF\_LGL\_CPCTY\_K = 6600020)       * For each of the above 4 cases, set respective flag to 1 (BLOCKED, UNDERINV, APAGOREYSH\_KINHSHS, THANONTAS) plus the FOUND\_OK flag, and exit for RC 4     - **If** something wrong appeared from previous checks (FOUND\_OK=1) **then**:       * Update table CRDT.EMBOS\_FILE\_DETAIL accordingly (reference flags: FOUND\_I\_IP=0, FOUND\_PERSON\_TYPE=0, FOUND\_ADDRESS=0, FOUND\_NAMES=0, BLOCKED=1, APAGOREYSH\_KINHSHS=1, THANONTAS=1, UNDERINV=1)     - If needed (DCLEMBOS\_FILE\_DETAIL.PE\_EMBOSSING\_NAME = ' ') update names for business products:       * Find Individual’s SYDIPEL (by calling SRVIXAS)       * Find Individual’s name (by calling SRVINMF) * **If** status is found to agree (FOUND\_OK=0), **then**:   + Update table CRDT.EMBOS\_FILE\_DETAIL |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER  KCCT.SMARTPIN  KCCT.SMARTPIN\_HIST |

#### [K04DP] Create files for PFM and for update of condition

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | K04DPOU1: File for PFM  DCL 1 RFTODPMS,  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME\_1 CHAR(30) INIT(' '),  2 CUSTNAME\_2 CHAR(30) INIT(' '),  2 EXP\_DATE CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '),  2 PVV CHAR(04) INIT(' '),  2 CVC2 CHAR(03) INIT(' '),  2 CVC1 CHAR(03) INIT(' '),  2 ADDRESS\_1 CHAR(30) INIT(' '),  2 ADDRESS\_2 CHAR(30) INIT(' '),  2 ACTION\_CODE CHAR(01) INIT(' '),  2 MAILER\_CODE CHAR(01) INIT(' '),  2 ENCRYPTED\_PIN CHAR(05) INIT(' '),  2 ERRORCODE CHAR(02) INIT(' '),  2 SPACE1 CHAR(01) INIT(' '),  2 SPACE2 CHAR(15) INIT(' '),  2 SPACE3 CHAR(15) INIT(' '),  2 SPACE4 CHAR(15) INIT(' '),  2 E\_PIN CHAR(01) INIT('Y'); |
| **OUTFILE2 FILE DECLARATION** | K04DPOU2: File for update of condition  DCL 1 RK04DPOU2,  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME CHAR(30) INIT(' '),  2 KIND CHAR(03) INIT(' '),  2 PIN\_BRANCH CHAR(03) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | IDBCCFT: Insert into KCCT.FILE\_TRAN |
| **Main Process Logic** | Retrieve current date.  Get file name from KCCT.FILE\_TRAN (by calling IDBCCFT).  Select NEW/PLA/PIN/MIG/RWL records from CRDT.EMBOS\_FILE\_DETAIL, and for every record: Write file for PFM. Write to file for update of condition.  Update file name and total records in KCCT.FILE\_TRAN. |
| **PROGRAM MAIN LOOP** | For every record returned from query to CRDT.EMBOS\_FILE\_DETAIL:   * Write to file 1 for PFM:   + Note: CUSTNAME only gets a value in case of cards with “on paper” PIN. In any other case the specific value isn’t used   + For product 041, translate EMBOSSING\_NAME   + Translate Street address and number, zipcode and city   + Set E\_PIN and MAILER\_CODE based on kind:     - (E\_PIN: Y=create new pin, N=don’t create new pin)     - (MAILER\_CODE: Y=print mailer, N=don’t print mailer)     - For NEW: set MAILER\_CODE to N, and E\_PIN to Y     - For PIN: set MAILER\_CODE to N, and E\_PIN to Y     - For PLA: set MAILER\_CODE to N, and E\_PIN to Y     - For RWL: set MAILER\_CODE to N, and E\_PIN to N     - For MIG: set MAILER\_CODE to N, and E\_PIN to Y     - Otherwise, do nothing   + Fill rest info, and write data to output file * Write to file 2 all records, for update of condition to 2 |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | KCCT.FILE\_TRAN  CRDT.EMBOS\_FILE\_DETAIL |

#### [K05DP] Update CRDT.EMBOS\_FILE\_DETAIL table condition from 1 to 2)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | K05DPIN1:  DCL 1 RK05DPIN1,  2 CARD CHAR(16),  2 CUSTNAME CHAR(30),  2 KIND CHAR(03),  2 PIN\_BRANCH CHAR(03); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_K05DPIN1: Raised when end of file is reached |
| **SERVICES** | SCDTRC: Select from KCCT.TRACK\_INFO  ICDTRC: Insert into KCCT.TRACK\_INFO |
| **Main Process Logic** | Retrieve current date.  For every line of input file: Update counters based on record kind, and trigger error for an unknown kind. Filter error cases. Update and insert data to tables. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * Update counter WS\_CNT\_FTOPMSPL for kind PIN * Update counter WS\_CNT\_FTOPMSPI for kind NEW/PLA/MIG/RWL * For any other kind: error * Retrieve card data form table CRDT.EMBOS\_FILE\_DETAIL * **If** card in wrong condition (still in step of collection) (DCLEMBOS\_FILE\_DETAIL.CONDITION = 0) **then** error * **Else If** card already sent for embossing (DCLEMBOS\_FILE\_DETAIL.CONDITION > 2) **then** error * **Else If** program has already run (DCLEMBOS\_FILE\_DETAIL.CONDITION = 2) **then** error * **Else**:   + Update CRDT.EMBOS\_FILE\_DETAIL   + Insert row into KCCT.TRACK\_INFO (by calling ICDTRC) |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER  KCCT.TRACK\_INFO |

#### [K06DP] Statistics from reading data

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | K06DPIN1:  DCL 1 RK06DPIN1,  2 CARD CHAR(16),  2 CUSTNAME CHAR(30),  2 KIND CHAR(03),  2 PIN\_BRANCH CHAR(03); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | K06DPOU1 (DUMMY) |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_K06DPIN1: Raised when end of file is reached |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve current date.  For every line of input file: For specified kinds, select info from table, and update variables/counters, ~~and write data to output file~~ (dummy file). |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * For kind NEW/PLA/MIG:   + Select details from CRDT.EMBOS\_FILE\_DETAIL   + **If** saved WS\_BRANCH is different from current record’s PIN\_BRANCH (from input file) **then**:     - Update saved WS\_BRANCH variable and counters     - ~~Print heading to file~~ (dummy file)   + ~~Write data to output file~~ (dummy file) * Otherwise no action |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER |

#### [K07DP] Update table and set condition form 2 to 3

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | K07DPIN1:  DCL 1 RK07DPIN1,  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME\_1 CHAR(30) INIT(' '),  2 CUSTNAME\_2 CHAR(30) INIT(' '),  2 EXP\_DATE CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '),  2 PVV CHAR(04) INIT(' '),  2 CVC2 CHAR(03) INIT(' '),  2 CVC1 CHAR(03) INIT(' '),  2 ICVV CHAR(03) INIT(' '),  2 PINBLOCK CHAR(16) INIT(' '),  2 ADDRESS\_1 CHAR(30) INIT(' '),  2 ADDRESS\_2 CHAR(30) INIT(' '),  2 ACTION\_CODE CHAR(01) INIT(' '),  2 MAILER\_CODE CHAR(01) INIT(' '),  2 ENCRYPTED\_PIN CHAR(05) INIT(' '),  2 ERROR\_CDE CHAR(02) INIT(' '),  2 SPACE1 CHAR(01) INIT(' '),  2 SPACE2 CHAR(15) INIT(' '),  2 SPACE3 CHAR(15) INIT(' '),  2 SPACE4 CHAR(15) INIT(' '),  2 E\_PIN CHAR(01) INIT('Y'),  2 MATRIX CHAR(15) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_K07DPIN1: Raised when end of file is reached |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve current date.  Initialize detail condition (Collect record with condition=2 and update them to 3): Select records form CRDT.EMBOS\_FILE\_DETAIL. Update records condition in CRDT.EMBOS\_FILE\_DETAIL to 2.  Check that input file has the right sum of records: Count records in input file. Find total records from KCCT.FILE\_TRAN. **If** they don’t agree, **then** error.  For every line of input file: Retrieve data from table, check for errors and compare to input file record. Update record condition to 3.  Check totals between file and records in table |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * Select embossing data from CRDT.EMBOS\_FILE\_DETAIL * **If** record is rejected (RK07DPIN1.ERROR\_CDE ^= '00'/’ ‘) **then** error * **If** record condition in CRDT.EMBOS\_FILE\_DETAIL is 3 (DCLEMBOS\_FILE\_DETAIL.CONDITION = 3) or not 2 (DCLEMBOS\_FILE\_DETAIL.CONDITION ^= 2), **then** error * **If** kind in file doesn’t agree with table (SUBSTR(RK07DPIN1.SPACE4,7,3) ^= DCLEMBOS\_FILE\_DETAIL.KIND), **then** error * **If** record kind in file is not NEW/PIN/PLA/MIG/RWL **then** error * Update record condition in CRDT.EMBOS\_FILE\_DETAIL to 3. |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER  KCCT.FILE\_TRAN |

#### [K08DP] Update table and set condition from 3 to 4

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | K08DPIN1:  DCL 1 RK08DPIN1,  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME\_1 CHAR(30) INIT(' '),  2 CUSTNAME\_2 CHAR(30) INIT(' '),  2 EXP\_DATE CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '),  2 PVV CHAR(04) INIT(' '),  2 CVC2 CHAR(03) INIT(' '),  2 CVC1 CHAR(03) INIT(' '),  2 ICVV CHAR(03) INIT(' '),  2 PINBLOCK CHAR(16) INIT(' '),  2 ADDRESS\_1 CHAR(30) INIT(' '),  2 ADDRESS\_2 CHAR(30) INIT(' '),  2 ACTION\_CODE CHAR(01) INIT(' '),  2 MAILER\_CODE CHAR(01) INIT(' '),  2 ENCRYPTED\_PIN CHAR(05) INIT(' '),  2 ERROR\_CDE CHAR(02) INIT(' '),  2 SPACE1 CHAR(01) INIT(' '),  2 SPACE2 CHAR(15) INIT(' '),  2 SPACE3 CHAR(15) INIT(' '),  2 SPACE4 CHAR(15) INIT(' '),  2 E\_PIN CHAR(01) INIT('Y'),  2 MATRIX CHAR(15) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_K08DPIN1: Raised when end of file is reached |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve the current date.  Check incomplete requests, by selecting records from CHECK\_ANEKPLHRWTA\_RTN.  For every line of input file: Retrieve information from CRDT.EMBOS\_FILE\_DETAIL. For PIN renewal, select information from CRDT.SMARTPIN\_TRACK. For detail condition 3, update CRDT.EMBOS\_FILE\_DETAIL, and for NEW/PIN/MIG/PLA/RWL update KCCT.SMARTPIN accordingly. For other condition, check for rejected cases. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * Retrieve information from CRDT.EMBOS\_FILE\_DETAIL * **If** E\_PIN is set (RK08DPIN1.E\_PIN = 'Y') and record kind is RWL(Renewal) (SUBSTR(RK08DPIN1.SPACE4,7,4) ^= 'RWL') **then** select data from CRDT.SMARTPIN\_TRACK * **If** detail condition 3 (from table) (DCLEMBOS\_FILE\_DETAIL.CONDITION = 3) **then**:   + Update CRDT.EMBOS\_FILE\_DETAIL   + **If** E\_PIN is set (RK08DPIN1.E\_PIN = 'Y') **then**:     - For record kind NEW/PIN/MIG/PLA:       * For record kind PIN (SUBSTR(RK08DPIN1.SPACE4,7,4) = 'PIN') with branch(DCLSMARTPIN\_TRACK.MATRIX\_SEND\_TYPE = '0'),   address(DCLSMARTPIN\_TRACK.MATRIX\_SEND\_TYPE = '1') or  email(DCLSMARTPIN\_TRACK.MATRIX\_SEND\_TYPE = '2') values:   * + - * + Select information from KCCT.SMARTPIN (for new reissue PIN request) for card, and for any non-zero (and non-100) sql code, error         + If no SQL code 100 appeared, then update record in KCCT.SMARTPIN (describe action to be performed next) and trigger error for any non-zero sql code       * Otherwise:         + Select information from KCCT.SMARTPIN for card, and for any non-zero (and non-100) sql code, error         + If no SQL code 100 appeared, then update record in KCCT.SMARTPIN with matrix         + Update record in CRDT.SMARTPIN\_TRACK to describe the operation to be performed next     - For record kind RWL no action     - For any other case, error * **Else if** condition is less than 2 (DCLEMBOS\_FILE\_DETAIL.CONDITION < 2) or greater than 4 (DCLEMBOS\_FILE\_DETAIL.CONDITION > 4) **then** error |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER  CRDT.SMARTPIN\_TRACK  KCCT.SMARTPIN |

#### [K09DP] Check not processed requests

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve the current date.  Collect all records in condition 1: Find number of records in CRDT.EMBOS\_FILE\_DETAIL that have kind value NEW/PLA/PIN/RWL/MIG with condition 1. For SQL code 100/-305 process must be run again. For any other (non-zero) SQL code, error. |
| **PROGRAM MAIN LOOP** | N/A |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### [K10DP] Create PAPYRUS file for current data

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | K10DPOU1:  DCL 1 RK10DPOU1,  2 CARD CHAR (16) INIT(' '),  2 FIL01 CHAR (01) INIT(' '),  2 REQUEST CHAR (01) INIT(' '), /\* '1' REISSUE,  '3' NEW \*/  2 FIL02 CHAR (01) INIT(' '),  2 REQ\_DATE CHAR (10) INIT(' '),  2 FIL03 CHAR (01) INIT(' '),  2 SND\_DATE CHAR (10) INIT(' '),  2 FIL04 CHAR (01) INIT(' '),  2 HOLDER CHAR (26) INIT(' '),  2 FIL05 CHAR (01) INIT(' '),  2 CUST\_NAME CHAR (40) INIT(' '),  2 FIL06 CHAR (01) INIT(' '),  2 COMP\_NAME CHAR (40) INIT(' '),  2 FIL07 CHAR (01) INIT(' '),  2 CRA\_ADDR CHAR (40) INIT(' '),  2 FIL08 CHAR (01) INIT(' '),  2 CRA\_TK CHAR (40) INIT(' '),  2 FIL09 CHAR (01) INIT(' '),  2 CRA\_CITY CHAR (40) INIT(' '),  2 FIL10 CHAR (01) INIT(' '),  2 CRA\_COUNTRY CHAR (40) INIT(' '),  2 FIL11 CHAR (01) INIT(' '),  2 MATRIX CHAR (15) INIT(' '),  2 FIL12 CHAR (01) INIT(' '),  2 FLAG\_ENT CHAR (02) INIT('NO'), /\* ΕΝΘΕΤΟ \*/  2 FIL13 CHAR (01) INIT(' '),  2 FLAG\_LET CHAR (02) INIT('NO'), /\* ΕΠΙΣΤΟΛΗ \*/  2 FIL14 CHAR (01) INIT(' '),  2 CARD\_ORIGIN CHAR (02) INIT(' '), /\* CC \*/  2 FIL15 CHAR (01) INIT(' '),  2 PRODUCT CHAR (09) INIT(' '),  2 FIL16 CHAR (01) INIT(' '),  2 PRODUCT\_NAME CHAR (30) INIT(' '),  2 FIL17 CHAR (01) INIT(' '),  2 CNT PIC '(6)9' INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | SRVINMF: Fetch names from CRAT.C\_IP\_X\_NM  SRVIXAL: Check account-customer relation  UCDSPT2: Update SEND\_STATUS, SEND\_TYPE & SEND\_DATE in CRDT.SMARTPIN\_TRACK |
| **Main Process Logic** | Retrieve the current date.  Select information  from KCCT.SMARTPIN for ”Reminder 15 Matrix” (REQUEST=1), smartpin delivery type“NBG branch” (FLG\_DELIVER=0) or “Mail to address” (FLG\_DELIVER=1), status “Inform from Q & PFM” (FLG=2), and product not (001000144) / "VISA PREPAID KEA W4" (000000144) / "VISA PREPAID ETE W4" (000000143) / "MC ΚΑΡΤΑ ΑΛΛΗΛΕΓΓΥΗΣ" (001000027) / "MC COSMOTE PREPAID" (001000033) / "ETHNOCASH PLUS" (000000101) / "ETHNOCASH PLUS BUSINESS" (000000102) / "MC DEBIT ΑΓΡΟΚΑΡΤΑ" (000000103) / "ETHNODEPOSIT" (000000104) / "MC DEBIT ΑΓΡΟΚΑΡΤΑ ΝΠ" (000000105) / "MC DEBIT CONSUMER" (000000106) / "MC DEBIT BUSINESS" (000000107) / "MC DEBIT SELF-EMPLOYED" (000000111) / "VISA DEBIT CONSUMER" (000000201) and from KCCT.SMARTPIN/CRDT.EMBOS\_FILE\_DETAIL for “SMARTPIN reissuance” (REQUEST=3), smartpin delivery type“At NBG store” (FLG\_DELIVER=0) or “Mail to address” (FLG\_DELIVER=1), status “Inform from Q & PFM” (FLG=2), and condition 4.  For every record returned from query: Update table to mask matrix. For delivery types other than NBG branch, retrieve customer data, and retrieve needed name based on specific case. Write data to output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every record of query:   * Update KCCT.SMARTPIN to mask matrix * If delivery type is not via NBG branch (RKCCSPIN.FLG\_DELIVER ^= '0') then:   + Retrieve customer data by calling SRVIXAL   + For product 000000041 (from KCCT.SMARTPIN):     - For legal entities (@SRVIXAL.ODATA.C\_IP\_TP=’ΝΠ’):       * Fetch distinctive title (WS\_NAME\_CLASS = 900004) from CRAT.C\_IP\_X\_NM (by calling SRVINMF)       * **If** not found, **then** fetch business title (WS\_NAME\_CLASS = 900003) from CRAT.C\_IP\_X\_NM (by calling SRVINMF)     - Otherwise (individual):       * Fetch distinctive title of sole proprietorship business (WS\_NAME\_CLASS = 900006) from CRAT.C\_IP\_X\_NM (by calling SRVINMF)       * **If** not found, **then** fetch first and last name (WS\_NAME\_CLASS = 900001) from CRAT.C\_IP\_X\_NM (by calling SRVINMF)     - **If** no names were found (from SRVINMF) **then** error     - Retrieve holder data (by calling SRVIXAL)     - Fetch names from CRAT.C\_IP\_X\_NM (by calling SRVINMF)     - **If** no names were found (from SRVINMF) **then** error   + Else:     - Fetch names from CRAT.C\_IP\_X\_NM (by calling SRVINMF)   + Write data to output file |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | KCCT.SMARTPIN  CRDT.SMARTPIN\_TRACK  CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER |

#### [K11DP] Send Email for SMARTPIN for reissued credit cards

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | CROSMS: Send message |
| **Main Process Logic** | Retrieve the current date.  Select information from KCCT.SMARTPIN/CRDT.EMBOS\_FILE\_DETAIL for SMARTPIN reissuance (REQUEST=3), delivery type email (FLG\_DELIVER=2), status “inform form Q & PFM” (FLG=2).  For every record returned from query: Send email, and update record as completed, and set matrix send status. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every record of query:   * Send email (by calling CROSMS) * In case of error rollback to before this execution * Otherwise   + Update KCCT.SMARTPIN, and set to completed (FLG=’1’), and in case of non-zero sql code, error   + Select details from CRDT.SMARTPIN\_TRACK   + Update CRDT.SMARTPIN\_TRACK (MATRIX\_SEND\_STATUS = '1' and MATRIX\_SEND\_DATE=current date) |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | KCCT.SMARTPIN  CRDT.SMARTPIN\_TRACK  CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER |

#### [K12DP] Write records to file for the update of PVV of cards in WAY4

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | K12DPOU1: File with card and PVV  DCL 1 O\_PINRESPONSE,  2 O\_CARD CHAR(19) INIT(' '),  2 O\_PVV CHAR(04) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | UCDEFD1: Update into CRDT.EMBOS\_FILE\_DETAIL |
| **Main Process Logic** | Select details from CRDT.EMBOS\_FILE\_DETAIL for any kind other than PIN, and for condition code 4.  For every record returned from query: Write NEW/PLA/MIG records in output file. Update record in table with condition 5. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every record of query:   * For records kind NEW/PLA/MIG:   + Write data to output file * Update record in CRDT.EMBOS\_FILE\_DETAIL with condition 5 (by calling UCDEFD1) |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### [K13DP] Create file with all PIN reissuance requests

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | K13DPOU1:  DCL 1 PINRESPONSE,  2 O\_WAY4FILENAME CHAR (040) INIT(' '),  2 O\_TRANSACTION\_ID CHAR (255) INIT(' '),  2 O\_BATCH\_ID CHAR (032) INIT(' '),  2 O\_PVV CHAR (004) INIT(' '),  2 O\_ERROR\_CDE CHAR (002) INIT(' '),  2 O\_CARD CHAR (019) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | UCDEFM1: Update into CRDT.EMBOS\_FILE\_MASTER  UCDEFD1: Update into CRDT.EMBOS\_FILE\_DETAIL (based on CARD)  UCDEFD3: Update into CRDT.EMBOS\_FILE\_DETAIL (based on FILE\_ID, CARD, KIND, MATURITY\_DATE, CONDITION)  SCDTRC: Select card from KCCT.TRACK\_INFO  ICDTRC: Insert into KCCT.TRACK\_INFO  SCDEFD3: Select details from CRDT.EMBOS\_FILE\_DETAIL and CRDT.EMBOS\_FILE\_MASTER (based on CARD PAN, EIDOS) |
| **Main Process Logic** | Retrieve current date.  Select details from CRDT.EMBOS\_FILE\_DETAIL for condition 4 and kind PIN.  For every record returned from query: Retrieve information from table, write data to output file, and update/insert into relevant tables. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every record of query:   * Retrieve card information (by calling SCDEFD3) * Write data to output file * Update record in CRDT.EMBOS\_FILE\_MASTER (by calling UCDEFM1) * Update record in CRDT.EMBOS\_FILE\_DETAIL (by calling UCDEFD1) (Mask PVV, and update condition to 4) * Update record in CRDT.EMBOS\_FILE\_DETAIL (by calling UCDEFD3) (Mask CVV1, CVV2, ICVV, I\_IP\_ET\_KAT, PINBLOCK and update condition to 12) * Retrieve latest track of card from KCCT.TRACK\_INFO (by calling SCDEFD3) * Insert record to KCCT.TRACK\_INFO (by calling ICDTRC) |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER  KCCT.TRACK\_INFO |

#### [K14DP] Create Embossing file to send to LYKOS

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | K14DPOU1: Embossing file for SSIS (Mastercard)  DCL 1 REMBOSC,  2 KINDCNT PIC'(06)9' INIT(0),  2 CC01 CHAR(01) INIT('{'),  2 CC02 CHAR(03) INIT('DCC'),  2 CC03 CHAR(01) INIT('{'),  2 CC04 CHAR(01) INIT('{'),  2 CC05 CHAR(03) INIT('EMB'),  2 CC06 CHAR(01) INIT('{'),  2 CARD\_4NUMBER CHAR(19) INIT(' '),  2 CC07 CHAR(01) INIT('}'),  2 FILLER01 CHAR(12) INIT(' '),  2 MM\_LHX CHAR(02) INIT(' '),  2 CC08 CHAR(01) INIT('/'),  2 YY\_LHX CHAR(02) INIT(' '),  2 FILLER02 CHAR(01) INIT(' '),  2 SEP01 CHAR(01) INIT('<'),  2 FILLER03 CHAR(08) INIT(' '),  2 CC09 CHAR(01) INIT('}'),  2 HOLDER CHAR(26) INIT(' '),  2 CC10 CHAR(01) INIT('}'),  2 COMPANY CHAR(26) INIT(' '),  2 CC11 CHAR(01) INIT('{'),  2 CC12 CHAR(03) INIT('OCR'),  2 CC13 CHAR(01) INIT('{'),  2 FILLER04 CHAR(12) INIT(' '),  2 CARD\_4LAST PIC'9999',  2 FILLER05 CHAR(01) INIT(' '),  2 CVV2\_INDENT PIC'999',  2 FILLER06 CHAR(01) INIT(' '),  2 CC14 CHAR(01) INIT('{'),  2 CC15 CHAR(03) INIT('ENC'),  2 CC16 CHAR(03) INIT('{%B'),  2 TRACK1\_CARD CHAR(16) INIT(' '),  2 CC17 CHAR(01) INIT('^'),  2 TRACK1\_NAME CHAR(26),  2 CC18 CHAR(01) INIT('^'),  2 YYMM\_LHXH\_1 PIC'9999',  2 SVC\_1 CHAR(03) INIT(' '),  2 CVV1\_1 PIC'999',  2 ZEROES\_10 CHAR(10) INIT('0000000000'),  2 CC19 CHAR(01) INIT('?'),  2 FILLER07 CHAR(01) INIT(' '),  2 CC20 CHAR(01) INIT(';'),  2 TRACK2\_CARD CHAR(16) INIT(' '),  2 CC21 CHAR(01) INIT('='),  2 YYMM\_LHXH\_2 PIC'9999',  2 SVC\_2 CHAR(03) INIT(' '),  2 CVV1\_2 PIC'999',  2 CARD\_SEQ\_NUM PIC'99' INIT('0'),  2 ZEROES\_1 CHAR(01) INIT('0'),  2 SVC\_3 CHAR(03) INIT(' '),  2 ZEROES\_4 CHAR(04) INIT('0000'),  2 CC22 CHAR(01) INIT('?'),  2 FILLER08 CHAR(01) INIT(' '),  2 CC23\_1 CHAR(01) INIT('6A'X), /\* | \*/  2 CC23\_2 CHAR(01) INIT('1'),  2 CC24 CHAR(02) INIT('\_0'),  2 CC25 CHAR(11) INIT('[0606 5060]'),  2 TEMPL\_NAME CHAR(12) INIT(' '),  2 CC26 CHAR(03) INIT('[1]'),  2 COMPANY\_NAME CHAR(35),  2 CC27 CHAR(03) INIT('[2]'),  2 FNAME CHAR(03),  2 CC28 CHAR(03) INIT('[3]'),  2 PNAME CHAR(01),  2 CC29 CHAR(03) INIT('[4]'),  2 SNAME CHAR(22),  2 CC30 CHAR(03) INIT('[5]'),  2 STREET CHAR(25),  2 CC31 CHAR(03) INIT('[6]'),  2 ZIPCODE CHAR(25),  2 CC32 CHAR(03) INIT('[7]'),  2 BRANCH CHAR(04),  2 CC33 CHAR(03) INIT('[8]'),  2 MAIL\_BARCODE CHAR(18),  2 CC34 CHAR(03) INIT('[A]'),  2 INSERT\_DATE CHAR(10) INIT(' '),  2 CC35 CHAR(03) INIT('[B]'),  2 PRODUCT CHAR(20) INIT(' '),  2 CC36 CHAR(03) INIT('[C]'),  2 CARD\_NUM\_TXT CHAR(08) INIT(' '),  2 CC37 CHAR(03) INIT('[D]'),  2 CREDIT\_LIMIT CHAR(07) INIT(' '),  2 FILLER\_09 CHAR(01) INIT(' '),  2 EURO\_01 CHAR(04) INIT('ΕΥΡΩ'),  2 FILLER\_10 CHAR(48) INIT(' '),  2 CC38 CHAR(03) INIT('[E]'),  2 WDRW\_LIMIT CHAR(07) INIT(' '),  2 FILLER\_11 CHAR(01) INIT(' '),  2 EURO\_02 CHAR(04) INIT('ΕΥΡΩ'),  2 FILLER\_12 CHAR(16) INIT(' '),  2 CC39 CHAR(03) INIT('[F]'),  2 MATRIX CHAR(16) INIT(' '),  2 CC40 CHAR(03) INIT('[G]'),  2 FUTURE CHAR(20) INIT(' '),  2 CC41 CHAR(03) INIT('[H]'),  2 LYK\_FIL\_NAME CHAR(10) INIT(' '),  2 CC42 CHAR(03) INIT('[K]'),  2 PAN\_MASKING CHAR(16) INIT(' '),  2 CC45 CHAR(03) INIT('[L]'),  2 APPL\_DATE CHAR(10) INIT(' '),  2 CC46 CHAR(03) INIT('[R]'),  2 MONTH\_VAL\_U CHAR(01) INIT(' '),  2 FILLER\_13 CHAR(39) INIT(' '),  2 FILLER\_14 CHAR(99) INIT(' '),  2 FILLER\_15 CHAR(05) INIT(' '),  2 CHIP\_CVC PIC'999',  2 FILLER\_16 CHAR(01) INIT(' '),  2 PLS\_DEL\_TYPE CHAR(01) INIT(' '),  2 OFFLINE\_PIN CHAR(01) INIT('1'),  2 XLS\_APPL CHAR(01) INIT('0'),  2 CC47 CHAR(01) INIT('~'),  2 FILLER\_17 CHAR(04) INIT(' '),  2 PIN\_BLOCK CHAR(16) INIT(' '),  2 FILLER\_18 CHAR(36) INIT(' '),  2 CC48 CHAR(01) INIT('`'),  2 FILLER\_19 CHAR(07) INIT(' '),  /\* MARK END OF CARD \*/  2 CC43 CHAR(05) INIT('{END{'),  2 CC44 CHAR(06) INIT('@@@@@@'); |
| **OUTFILE2 FILE DECLARATION** | K14DPOU2: Embossing file for SSIS (VISA)  DCL 1 REMBOSCV,  2 KINDCNT PIC'(06)9' INIT(0),  2 CC01 CHAR(01) INIT('{'),  2 CC02 CHAR(03) INIT('DCC'),  2 CC03 CHAR(01) INIT('{'),  2 CC04 CHAR(01) INIT('{'),  2 CC05 CHAR(03) INIT('EMB'),  2 CC06 CHAR(01) INIT('{'),  2 CARD\_4NUMBER CHAR(19) INIT(' '),  2 CC07 CHAR(01) INIT('}'),  2 FILLER01 CHAR(12) INIT(' '),  2 MM\_LHX CHAR(02) INIT(' '),  2 CC08 CHAR(01) INIT('/'),  2 YY\_LHX CHAR(02) INIT(' '),  2 FILLER02 CHAR(01) INIT(' '),  2 SEP01 CHAR(01) INIT('<'),  2 FILLER03 CHAR(08) INIT(' '),  2 CC09 CHAR(01) INIT('}'),  2 HOLDER CHAR(26) INIT(' '),  2 CC10 CHAR(01) INIT('}'),  2 COMPANY CHAR(26) INIT(' '),  2 CC11 CHAR(01) INIT('{'),  2 CC12 CHAR(03) INIT('OCR'),  2 CC13 CHAR(01) INIT('{'),  2 FILLER04 CHAR(12) INIT(' '),  2 CARD\_4LAST PIC'9999',  2 FILLER05 CHAR(01) INIT(' '),  2 CVV2\_INDENT PIC'999',  2 FILLER06 CHAR(01) INIT(' '),  2 CC14 CHAR(01) INIT('{'),  2 CC15 CHAR(03) INIT('ENC'),  2 CC16 CHAR(03) INIT('{%B'),  2 TRACK1\_CARD CHAR(16) INIT(' '),  2 CC17 CHAR(01) INIT('^'),  2 TRACK1\_NAME CHAR(26),  2 CC18 CHAR(01) INIT('^'),  2 YYMM\_LHXH\_1 PIC'9999',  2 SVC\_1 CHAR(03) INIT(' '),  2 DISCRET CHAR(13) INIT(' ') ,  2 ZEROES\_02\_V1 CHAR(02) INIT('00') ,  2 CVV1\_1 PIC'999' ,  2 ZEROES\_02\_V2 CHAR(02) INIT('00') ,  2 ACI CHAR(01) INIT(' ') ,  2 ZEROES\_03\_V1 CHAR(03) INIT('000'),  2 CC19 CHAR(01) INIT('?'),  2 FILLER07 CHAR(01) INIT(' '),  2 CC20 CHAR(01) INIT(';'),  2 TRACK2\_CARD CHAR(16) INIT(' '),  2 CC21 CHAR(01) INIT('='),  2 YYMM\_LHXH\_2 PIC'9999',  2 SVC\_2 CHAR(03) INIT(' '),  2 CVV1\_2 PIC'999',  2 CARD\_SEQ\_NUM PIC'99' INIT('0'),  2 ZEROES\_1 CHAR(01) INIT('0'),  2 SVC\_3 CHAR(03) INIT(' '),  2 ZEROES\_4 CHAR(04) INIT('0000'),  2 CC22 CHAR(01) INIT('?'),  2 FILLER08 CHAR(01) INIT(' '),  2 CC23\_1 CHAR(01) INIT('6A'X),  2 CC23\_2 CHAR(01) INIT('1'),  2 CC24 CHAR(02) INIT('\_0'),  2 CC25 CHAR(11) INIT('[0606 5060]'),  2 TEMPL\_NAME CHAR(12) INIT(' '),  2 CC26 CHAR(03) INIT('[1]'),  2 COMPANY\_NAME CHAR(35),  2 CC27 CHAR(03) INIT('[2]'),  2 FNAME CHAR(03),  2 CC28 CHAR(03) INIT('[3]'),  2 PNAME CHAR(01),  2 CC29 CHAR(03) INIT('[4]'),  2 SNAME CHAR(22),  2 CC30 CHAR(03) INIT('[5]'),  2 STREET CHAR(25),  2 CC31 CHAR(03) INIT('[6]'),  2 ZIPCODE CHAR(25),  2 CC32 CHAR(03) INIT('[7]'),  2 BRANCH CHAR(04),  2 CC33 CHAR(03) INIT('[8]'),  2 MAIL\_BARCODE CHAR(18),  2 CC34 CHAR(03) INIT('[A]'),  2 INSERT\_DATE CHAR(10) INIT(' '),  2 CC35 CHAR(03) INIT('[B]'),  2 PRODUCT CHAR(20) INIT(' '),  2 CC36 CHAR(03) INIT('[C]'),  2 CARD\_NUM\_TXT CHAR(08) INIT(' '),  2 CC37 CHAR(03) INIT('[D]'),  2 CREDIT\_LIMIT CHAR(07) INIT(' '),  2 FILLER\_09 CHAR(01) INIT(' '),  2 EURO\_01 CHAR(04) INIT('ΕΥΡΩ'),  2 FILLER\_10 CHAR(48) INIT(' '),  2 CC38 CHAR(03) INIT('[E]'),  2 WDRW\_LIMIT CHAR(07) INIT(' '),  2 FILLER\_11 CHAR(01) INIT(' '),  2 EURO\_02 CHAR(04) INIT('ΕΥΡΩ'),  2 FILLER\_12 CHAR(16) INIT(' '),  2 CC39 CHAR(03) INIT('[F]'),  2 MATRIX CHAR(16) INIT(' '),  2 CC40 CHAR(03) INIT('[G]'),  2 FUTURE CHAR(20) INIT(' '),  2 CC41 CHAR(03) INIT('[H]'),  2 LYK\_FIL\_NAME CHAR(10) INIT(' '),  2 CC42 CHAR(03) INIT('[K]'),  2 PAN\_MASKING CHAR(16) INIT(' '),  2 CC45 CHAR(03) INIT('[L]'),  2 APPL\_DATE CHAR(10) INIT(' '),  2 CC46 CHAR(03) INIT('[R]'),  2 MONTH\_VAL\_U CHAR(01) INIT(' '),  2 FILLER\_13 CHAR(39) INIT(' '),  2 FILLER\_14 CHAR(99) INIT(' '),  2 FILLER\_15 CHAR(05) INIT(' '),  2 CHIP\_CVC PIC'999',  2 FILLER\_16 CHAR(01) INIT(' '),  2 PLS\_DEL\_TYPE CHAR(01) INIT(' '),  2 OFFLINE\_PIN CHAR(01) INIT('1'),  2 XLS\_APPL CHAR(01) INIT('0'),  2 CC47 CHAR(01) INIT('~'),  2 FILLER\_17 CHAR(04) INIT(' '),  2 PIN\_BLOCK CHAR(16) INIT(' '),  2 FILLER\_18 CHAR(36) INIT(' '),  2 CC48 CHAR(01) INIT('`'),  2 FILLER\_19 CHAR(07) INIT(' '),  2 CC43 CHAR(05) INIT('{END{'),  2 CC44 CHAR(06) INIT('@@@@@@'); |
| **OUTFILE3 FILE DECLARATION** | K14DPOU3: Embossing file for SSIS (All - Mastercard & VISA)  DCL 1 REMBOALL,  2 KINDCNT PIC '(06)9' INIT(0),  2 HOLDER CHAR (026) INIT(' '),  2 COMPANY CHAR (026) INIT(' '),  2 TRACK1\_CARD CHAR (016) INIT(' '),  2 STREET CHAR (025) INIT(' '),  2 ZIPCODE CHAR (005) INIT(' '),  2 CITY CHAR (010) INIT(' '),  2 FILE\_EMB\_NAM CHAR (003) INIT(' '),  2 FILE\_EMB\_CNT PIC '99999' INIT(' '),  2 PLS\_DEL\_TYPE CHAR (001) INIT(' '); |
| **OUTFILE4 FILE DECLARATION** | K14DPOU4: File for AS400  DCL 1 AS400REC,  2 KINDCNT PIC'(06)9' INIT(0),  2 CC01 CHAR(01) INIT('{'),  2 CC02 CHAR(03) INIT('DCC'),  2 CC03 CHAR(01) INIT('{'),  2 CC04 CHAR(01) INIT('{'),  2 CC05 CHAR(03) INIT('EMB'),  2 CC06 CHAR(01) INIT('{'),  2 LINE1\_CARD CHAR(19) INIT(' '),  2 CC07 CHAR(01) INIT('}'),  2 FILLER01 CHAR(12) INIT(' '),  2 MM\_LHX CHAR(02) INIT(' '),  2 CC08 CHAR(1) INIT('/'),  2 YY\_LHX CHAR(02) INIT(' '),  2 FILLER02 CHAR(10) INIT(' '),  2 CC09 CHAR(01) INIT('}'),  2 HOLDER CHAR(26) INIT(' '),  2 CC10 CHAR(01) INIT('}'),  2 COMPANY CHAR(26) INIT(' '),  2 CC11 CHAR(01) INIT('{'),  2 CC12 CHAR(03) INIT('OCR'),  2 CC13 CHAR(01) INIT('{'),  2 FILLER03 CHAR(12) INIT(' '),  2 CARD\_4LAST PIC'9999',  2 FILLER04 CHAR(01) INIT(' '),  2 CVV2\_INDENT PIC'999',  2 FILLER05 CHAR(01) INIT(' '),  2 CC14 CHAR(01) INIT('{'),  2 CC15 CHAR(03) INIT('ENC'),  2 CC16 CHAR(03) INIT('{%B'),  2 TRACK1\_CARD CHAR(16) INIT(' '),  2 CC17 CHAR(01) INIT('^'),  2 TRACK1\_NAME CHAR(26),  2 CC18 CHAR(01) INIT('^'),  2 YYMM\_LHXH\_1 PIC'9999',  2 SVC\_1 CHAR(03) INIT(' '),  2 CVV1\_1 PIC'999',  2 ZEROES\_10 CHAR(10) INIT('0000000000'),  2 CC19 CHAR(01) INIT('?'),  2 FILLER06 CHAR(01) INIT(' '),  2 CC20 CHAR(01) INIT(';'),  2 TRACK2\_CARD CHAR(16) INIT(' '),  2 CC21 CHAR(01) INIT('='),  2 YYMM\_LHXH\_2 PIC'9999',  2 SVC\_2 CHAR(03) INIT(' '),  2 CVV1\_2 PIC'999',  2 CARD\_SEQ\_NUM PIC'99' INIT('0'),  2 ZEROES\_8 CHAR(08) INIT('00000000'),  2 CC22 CHAR(01) INIT('?'),  2 FILLER07 CHAR(01) INIT(' '),  2 CC23 CHAR(002) INIT('|1'),  2 CC24 CHAR(02) INIT('\_0'),  2 CC25 CHAR(11) INIT('[4037 0883]'),  2 TEMPL\_NAME CHAR(12) INIT(' '),  2 CC26 CHAR(03) INIT('[1]'),  2 KINDCNT2 CHAR(06),  2 CC27 CHAR(03) INIT('[2]'),  2 BRANCH CHAR(03),  2 CC28 CHAR(03) INIT('[3]'),  2 SNAME CHAR(22),  2 CC29 CHAR(03) INIT('[4]'),  2 PNAME CHAR(01),  2 CC30 CHAR(03) INIT('[5]'),  2 FNAME CHAR(03),  2 CC31 CHAR(03) INIT('[6]'),  2 STREET CHAR(25),  2 CC32 CHAR(03) INIT('[7]'),  2 ZIPCODE CHAR(05),  2 CC33 CHAR(03) INIT('[8]'),  2 CITY CHAR(10),  2 CC34 CHAR(03) INIT('[9]'),  2 HMER CHAR(10) INIT(' '),  2 CC35 CHAR(03) INIT('[A]'),  2 POS\_LIMIT CHAR (009),  2 CC36 CHAR(03) INIT('[B]'),  2 WDRW\_LIMIT CHAR (009),  2 CC37 CHAR(03) INIT('[C]'),  2 FILE\_EMB\_NAM CHAR (003) INIT(' '),  2 FILE\_EMB\_CNT PIC '99999',  2 FILLER08 CHAR(19) INIT(' '),  2 ICVV\_USER PIC'999',  2 PIN\_BLOCK CHAR(16) INIT(' '),  2 OFFLINEPIN CHAR(01) INIT(' '),  2 XLS CHAR(01) INIT(' '),  2 FILLER09 CHAR(03) INIT(' '),  2 PRODUCTCODE CHAR(03) INIT(' '),  2 PROCCDCODE CHAR(02) INIT(' '),  2 IDENTITY CHAR(09) INIT(' '),  2 REQUEST\_DATE CHAR(10) INIT(' '),  2 FILLER10 CHAR(01) INIT(' '),  2 INT\_LIMIT CHAR (009),  2 MATRIX CHAR(15) INIT(' '),  2 BIRTHDATE CHAR(10) INIT(' '),  2 I\_IP CHAR(10) INIT(' '),  2 COUNTRY CHAR(16) INIT(' '),  2 CC38 CHAR(05) INIT('{END{'),  2 CC39 CHAR(06) INIT('@@@@@@'); |
| **OUTFILE5 FILE DECLARATION** | N/A (DUMMY) |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | UCDEFD3: Update into CRDT.EMBOS\_FILE\_DETAIL (based on FILE\_ID, CARD, KIND, MATURITY\_DATE, CONDITION)  SCDEPM: Select from CRDT.EMBOS\_PARM  SCDECR: Select from CRDT.EMBOS\_COUNTER  UCDECR: Update into CRDT.EMBOS\_COUNTER  SCCSPN1: Select from KCCT.SMARTPIN  UCCSPN1: Update into KCCT.SMARTPIN  UCDEFM2: Update into CRDT.EMBOS\_FILE\_MASTER |
| **Main Process Logic** | Retrieve current date.  Select details from CRDT.EMBOS\_FILE\_DETAIL/CRDT.EMBOS\_FILE\_MASTER for NEW/PLA/MIG/RWL and condition 5.  For every record returned from query: Run validations, write to files, and update tables.  If LYKOS status count (LYKOS\_STATUS\_CNT) is greater than 0, then update CRDT.EMBOS\_FILE\_MASTER (by calling UCDEFM2) |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every record of query:   * For non-RWL kind requests (plastic replacement requests) (DCLEMBOS\_FILE\_DETAIL.KIND ^= 'RWL'):   + Retrieve details from KCCT.SMARTPIN (by calling SCCSPN1)   + Mask matrix and update KCCT.SMARTPIN (by calling UCCSPN1)   + For delivery type SMS (@SCCSPN1.ODATA.FLG\_DELIVER = '3') and request status “New issuance with SMS” (@SCCSPN1.ODATA.FLG = '4'), do nothing   + Otherwise, update FLG=’1’ (completed) and timestamp in KCCT.SMARTPIN. * **If** previous WAY4 file name does not agree with current name (PREV\_WAY4\_FILE\_NAME != DCLEMBOS\_FILE\_MASTER.WAY4\_FILE\_NAME) **then**:   + **If** LYKOS status count is greater than 0 (LYKOS\_STATUS\_CNT > 0), **then**:     - Update CRDT.EMBOS\_FILE\_MASTER (by calling UCDEFM2)   + Retrieve file name from CRDT.EMBOS\_PARM (by calling SCDEPM)   + Retrieve last counter used from CRDT.EMBOS\_COUNTER (by calling SCDECR)   + Update last counter used in CRDT.EMBOS\_COUNTER (by calling UCDECR) * Increase LYKOS status counter (LYKOS\_STATUS\_CNT) * Process iteration data and write either to embossing file, ~~or to CMOD file~~ (dummy)   + For Mastercard (MC) detail scheme (DCLEMBOS\_FILE\_DETAIL.SCHEME):     - **If** production event is not instant (DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT != 'INS') **then**:       * Format output data (all cases except instant)       * Format output data from AS400 rec     - **Else**:       * Format output data (for instant case)       * Format output data from AS400 rec   + For VISA detail scheme (DCLEMBOS\_FILE\_DETAIL.SCHEME):     - **If** production event is not instant (DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT != 'INS') **then**:       * Format output data (all cases except instant)       * Format output data from AS400 rec     - **Else**:       * Format output data (for instant case)       * Format output data from AS400 rec   + Otherwise, scheme not recognised   + Perform Validations (if any fails, skip the rest)     - Validate Embossing:       * For Consumer(DCLEMBOS\_FILE\_DETAIL.PRODUCT\_GROUP=’06’) / Business(DCLEMBOS\_FILE\_DETAIL.PRODUCT\_GROUP=’07’)         + Embossing name must not be empty       * Otherwise error     - Validate Expiration Date (not empty, in format DD.MM.YYYY, DD>=01, MM<=12, YYYY>=2015)     - Validate SVC:       * Product Code (DCLEMBOS\_FILE\_DETAIL.PRODUCT\_CODE) must be '021'/'023'/'025'/'041'/'067'/'117'/'121'/'151'         + SVC (DCLEMBOS\_FILE\_DETAIL.SVC) must not be ‘206’       * Otherwise error     - Validate SNAME: for not Instant cases (DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT ^= 'INS'), DCLEMBOS\_FILE\_DETAIL.SNAME can’t be empty     - Validate street: for not Instant cases (DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT ^= 'INS'), DCLEMBOS\_FILE\_DETAIL.STREET can’t be empty     - Validate Plastic delivery type:       * Delivery types (DCLEMBOS\_FILE\_DETAIL.PL\_DELIVERY\_TYPE) other than 02(Email), 04(Voice), 07(), 09() are invalid       * For kind (DCLEMBOS\_FILE\_DETAIL.KIND) NEW/MIG/PLA, for any other product type (DCLEMBOS\_FILE\_DETAIL.PRODUCT\_CODE) than 041, delivery types (DCLEMBOS\_FILE\_DETAIL.PL\_DELIVERY\_TYPE) other than 02(Email), 04(Voice), 07(), 09() are invalid       * For kind (DCLEMBOS\_FILE\_DETAIL.KIND) NEW/MIG/PLA, for any other product type (DCLEMBOS\_FILE\_DETAIL.PRODUCT\_CODE) than 041, delivery types (DCLEMBOS\_FILE\_DETAIL.PL\_DELIVERY\_TYPE) other than 02(Email), 07() are invalid       * For RWL kind (DCLEMBOS\_FILE\_DETAIL.KIND), delivery types (DCLEMBOS\_FILE\_DETAIL.PL\_DELIVERY\_TYPE) other than 02(Email), 04(Voice), 07(), are invalid     - Validate PIN block: PIN block (DCLEMBOS\_FILE\_DETAIL.PIN\_BLOCK) can’t be empty     - Validate cardholder card number: card number must only contain digits 0-9, and spaces, and if any of the cardholder’s card number quadruplets is empty, card number is invalid     - Validate branch: for not Instant cases (DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT ^= 'INS'), branch code can’t be empty (SUBSTR(DCLEMBOS\_FILE\_DETAIL.PIN\_DELIVERY\_BRANCH,1,3))     - Validate insertion date: for not Instant cases (DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT ^= 'INS'), issuing date (DCLEMBOS\_FILE\_DETAIL.ISUUING\_DATE) can’t be empty     - Validate daily credit limit: for not Instant cases (DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT ^= 'INS'), POS purchase limit (DCLEMBOS\_FILE\_DETAIL.POS\_PURCHASE\_LMT) can’t be 0     - Validate daily withdrawal limit: for not Instant cases (DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT ^= 'INS'), daily withdrawal limit (DCLEMBOS\_FILE\_DETAIL.TOTAL\_CASH\_WTH\_LMT) can’t be empty     - Validate file embossing name: for not Instant cases (DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT ^= 'INS'), LYKOS prefix (L\_EMBOS\_PARM.LYKOS\_PREFIX) can’t be empty     - Validate file cardholder name/track1 name/company name:       * For not Instant cases (DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT ^= 'INS')         + For Consumer (DCLEMBOS\_FILE\_DETAIL.PRODUCT\_GROUP=’06’): Check value of holder (A-Z, 0-9, spaces, and special characters) and track1 name (translate and make usus it uses A-Z, 0-9, spaces, and special characters)         + For Business (DCLEMBOS\_FILE\_DETAIL.PRODUCT\_GROUP=’07’):   **If** PE\_EMBOSSING\_NAME is empty (NOTE: error case, remains for migration purposes) **then**:  Verify embossing name and track1  **Else**:  Verify track1 and holder name   * + - * + Otherwise, error   + **If** no error from validations, **then** write data to output:     - Evaluate record ID:       * WS\_TYPE = DCLEMBOS\_FILE\_MASTER.PRODUCTION\_CODE!!DCLEMBOS\_FILE\_MASTER.PRODUCT\_CODE       * Acceptable WS\_TYPE values:       * 'A021','AA21' (NEW ISSUANCE - MASTERCARD CLASSIC 527890) 'A023','AA23' (NEW ISSUANCE - MASTERCARD GOLD)   'A025','AA25' (NEW ISSUANCE - MASTERCARD PLATINUM) 'A041' (NEW ISSUANCE - MASTERCARD BUSINESS) 'A067' (NEW ISSUANCE - MASTERCARD CLASSIC 527801) 'A117' (NEW ISSUANCE - VISA IASO) 'A121' (NEW ISSUANCE - VISA CLASSIC) 'A151' (NEW ISSUANCE - VISA TOYOTA) 'B021','BA21' (PLASTIC REPLACEMENT - MASTERCARD CLASSIC 527890) 'B023','BA23' (PLASTIC REPLACEMENT - MASTERCARD GOLD) 'B025','BA25' (PLASTIC REPLACEMENT - MASTERCARD PLATINUM) 'B041' (PLASTIC REPLACEMENT - MASTERCARD BUSINESS) 'B067' (PLASTIC REPLACEMENT - MASTERCARD CLASSIC 527801) 'B117' (PLASTIC REPLACEMENT - VISA IASO) 'B121' (PLASTIC REPLACEMENT - VISA CLASSIC) 'B151' (PLASTIC REPLACEMENT - VISA TOYOTA) 'C021','CA21' (MIGRATION - MASTERCARD CLASSIC 527890) 'C023','CA23' (MIGRATION - MASTERCARD GOLD) 'C025','CA25' (MIGRATION - MASTERCARD PLATINUM) 'C041' (MIGRATION - MASTERCARD BUSINESS) 'C067' (MIGRATION - MASTERCARD CLASSIC 527801) 'C117' (MIGRATION - VISA IASO) 'C121' (MIGRATION - VISA CLASSIC) 'C151' (MIGRATION - VISA TOYOTA) 'E021','EA21' (RENEWAL - MASTERCARD CLASSIC 527890) 'E023','EA23' (RENEWAL - MASTERCARD GOLD) 'E025','EA25' (RENEWAL - MASTERCARD PLATINUM) 'E041' (RENEWAL - MASTERCARD BUSINESS) 'E067' (RENEWAL - MASTERCARD CLASSIC 527801) 'E117' (RENEWAL - VISA IASO) 'E121' (RENEWAL - VISA CLASSIC) 'E151' (RENEWAL - VISA TOYOTA)   * + - * For any other case, error     - Write data to embossing file sent to LYKOS:       * For Mastercard (DCLEMBOS\_FILE\_DETAIL.SCHEME=’MC’):         + Write to LYKOS file (Mastercard)         + Write to AS400 file       * For VISA (DCLEMBOS\_FILE\_DETAIL.SCHEME=’VISA’):         + Write to LYKOS file (VISA)         + Write to AS400 file       * Otherwise, error     - Write output to LYKOS file (All) For further process   + ~~Else write data to CMOD file~~ (dummy) * Update CRDT.EMBOS\_FILE\_DETAIL with encryption characters (by calling UCDEFD3) * Hold file name (DCLEMBOS\_FILE\_MASTER.WAY4\_FILE\_NAME) to previous file name local variable |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER  KCCT.SMARTPIN  CRDT.EMBOS\_PARM  CRDT.EMBOS\_COUNTER |

#### [K15DP] Process file to be sent to SSIS and then to LYKOS

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | K15DPIN1:  DCL 1 REMBOS,  2 KINDCNT PIC '(06)9' INIT(0),  2 HOLDER CHAR (026) INIT(' '),  2 COMPANY CHAR (026) INIT(' '),  2 TRACK1\_CARD CHAR (016) INIT(' '),  2 STREET CHAR (025) INIT(' '),  2 ZIPCODE CHAR (005) INIT(' '),  2 CITY CHAR (010) INIT(' '),  2 FILE\_EMB\_NAM CHAR (003) INIT(' '),  2 FILE\_EMB\_CNT PIC '99999' INIT(' '),  2 PLS\_DEL\_TYPE CHAR (001) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | READ\_K15DPIN1: Raised when end of file is reached |
| **SERVICES** | ICDTRC: Insert into KCCT.TRACK\_INFO  SCDEFD2: Select from CRDT.EMBOS\_FILE\_DETAIL  SCDTRC: Select from KCCT.TRACK\_INFO (last card) |
| **Main Process Logic** | Retrieve current date.  For every line of input file: Retrieve details from tables, and insert new records. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * Retrieve details from CRDT.EMBOS\_FILE\_DETAIL (by calling SCDEFD2) * Retrieve latest track of card (by calling SCDTRC) * Insert record into KCCT.TRACK\_INFO (by calling ICDTRC) and commit |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | KCCT.TRACK\_INFO  CRDT.EMBOS\_FILE\_DETAIL |

#### [K17DP] Update table EMBOS\_FILE\_DETAIL and set condition form 5 to 6

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | K17DPIN1:  DCL 1 RK17DPIN1,  2 FILLER1 CHAR(058) INIT(' '),  2 CARD CHAR(016) INIT(' '),  2 FILLER2 CHAR(049) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_K17DPIN1: Raised when end of file is reached |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve current date.  For every line of input file: Handle error cases, and update card records when needed. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * Select card details from CRDT.EMBOS\_FILE\_DETAIL * **If** card is anonymous (DCLEMBOS\_FILE\_DETAIL.CONDITION < 5) **then**: error * **Else If** card is rejected (DCLEMBOS\_FILE\_DETAIL.CONDITION > 6) **then**: error * **Else If** program has run before, and table record was updated with condition 6 (DCLEMBOS\_FILE\_DETAIL.CONDITION = 6), **then**: print error message, and continue * **Else**    + Update record in CRDT.EMBOS\_FILE\_DETAIL with condition 6   + Commit records |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### [K18DP] Write data to dummy file

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | K18DPOU1 (DUMMY) |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve current date.  Select records from CRDT.EMBOS\_FILE\_DETAIL for kind NEW/PLA/PIN/MIG/RWL, with condition 8/9.  For every record of query: Update issuing branch local variable, ~~and write data to output file~~ (DUMMY) |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every record of query:   * **If** saved WS\_BRANCH doesn’t agree with query’s issuing branch (WS\_BRANCH != SUBSTR(DCLEMBOS\_FILE\_DETAIL.ISSUING\_BRANCH,1,3)) **then**:   + Update local WS\_BRANCH variable with issuing branch value   + ~~Write headings to file~~ (DUMMY) * ~~Write data to output file~~ (DUMMY) |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### [K19DP] Create CMOD statistics file regarding LYKOS daily file

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | K19DPOU1 (DUMMY) |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve current date.  Select count of records per issuing branch and branch from CRDT.EMBOS\_FILE\_DETAIL for kind NEW/PLA/MIG/RWL in the past 3 hours with condition 6.  For every record of query: Find branch and counts, evaluate counts, and write data to output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every record of query:   * Find branch title from DXPT.GEOST * Fetch count for every product code and kind combo from the last 3 hours, where kind is MIG/NEW/PLA/RWL, and condition is 6. * For every record:   + Evaluate result of fetched count based on kind and product:     - L\_KDPR (local variable) = L\_FKND || L\_FPRD (concatenation)     - Update specified counter variable with count for following values of L\_KDPR:   MIG021  MIG023  MIG025  MIG041  MIG067  MIG117  MIG121  MIG151  NEW021  NEW023  NEW025  NEW041  NEW067  NEW117  NEW121  NEW151  PLA021  PLA023  PLA025  PLA041  PLA067  PLA117  PLA121  PLA151  RWL021  RWL023  RWL025  RWL041  RWL067  RWL117  RWL121  RWL151   * ~~Write data to output file~~ (DUMMY) with total card counts   + Write header: (CRD053-DPL)   + Write rows for products     - 021 (Mastercard classic)       * 'MASTERCARD CLASSIC',   '021',  L\_NEWCNT021 + L\_NEWCNT067,  L\_MIGCNT021 + L\_MIGCNT067,  L\_PLACNT021 + L\_PLACNT067,  L\_RWLCNT021 + L\_RWLCNT067   * + - 023 (Mastercard gold)       * 'MASTERCARD GOLD',   '023',  L\_NEWCNT023,  L\_MIGCNT023,  L\_PLACNT023,  L\_RWLCNT023   * + - 025 (Mastercard Platinum)       * 'MASTERCARD PLATINUM',   '025',  L\_NEWCNT025,  L\_MIGCNT025,  L\_PLACNT025,  L\_RWLCNT025   * + - 041 (Mastercard Business)       * 'MASTERCARD BUSINESS',   '041',  L\_NEWCNT041,  L\_MIGCNT041,  L\_PLACNT041,  L\_RWLCNT041   * + - 117 (VISA IASO)       * 'VISA IASO',   '117',  L\_NEWCNT117,  L\_MIGCNT117,  L\_PLACNT117,  L\_RWLCNT117   * + - 121 (VISA Classic)       * 'VISA CLASSIC'   '121',  L\_NEWCNT121,  L\_MIGCNT121,  L\_PLACNT121,  L\_RWLCNT121   * + - 151 (VISA TOYOTA)       * 'VISA TOYOTA',   '151',  L\_NEWCNT151,  L\_MIGCNT151,  L\_PLACNT151,  L\_RWLCNT151 |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### [K20DP] Collect rejected cards, Send response to SSIS (to be sent to WAY4)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | K20DPOU1:  DCL 1 RK20DPOU1,  2 O\_REC\_NO CHAR (06) INIT(' '),  2 O\_CARD\_EMB CHAR (26) INIT(' '),  2 O\_CARD\_NO CHAR (16) INIT(' '),  2 O\_FILLER CHAR (02) INIT(' '),  2 O\_REJ\_CODE CHAR (03) INIT(' '),  2 O\_REJ\_DESCR CHAR (100) INIT(' '),  2 O\_FILE CHAR (15) INIT(' '),  2 O\_ELTA\_DATE CHAR (10) INIT(' '),  2 O\_ORIG\_REC\_ID CHAR (06) INIT(' '),  2 O\_EXCEPT\_FLG CHAR (01) INIT(' '),  2 O\_PLS\_DEL\_TYPE CHAR (02) INIT(' '),  2 WAY4\_FILE\_NAME CHAR (40) INIT(' '),  2 TRANSACTION\_ID CHAR (255) INIT(' '),  2 BATCH\_ID CHAR (32) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | K20DPOU2:  DCL 1 RK20DPOU2,  2 CARD CHAR (16) INIT(' '),  2 FILLER\_1 CHAR (01) INIT(';'),  2 SYDIPEL CHAR (11) INIT(' '),  2 FILLER\_2 CHAR (01) INIT(';'),  2 CLASSIFIER\_VALUE CHAR (01) INIT(' '),  2 FILLER\_3 CHAR (01) INIT(';'),  2 CLASSIFIER\_NAME CHAR (16) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | UCDEFD2: Update into CRDT.EMBOS\_FILE\_DETAIL  ~~SRVIXAS: Retrieve card holder’s SYDIPEL (active relation)~~ |
| **Main Process Logic** | Retrieve current date.  Select records from CRDT.EMBOS\_FILE\_DETAIL / CRDT.EMBOS\_FILE\_MASTER for records status 8/9/10/14 and condition 8/9/10/14.  For every record of query: Evaluate rejections and write data to output file 1. For renewals, write data to output file 2. Update table CRDT.EMBOS\_FILE\_DETAIL. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every record of query:   * Evaluate, and write rejections to file 1:   + **If** condition is 8 (DCLEMBOS\_FILE\_DETAIL.CONDITION = 8) **then**: error code/text 008/MISSING SIDIPEL DATA   + **ELse If** condition is 9 (DCLEMBOS\_FILE\_DETAIL.CONDITION = 9) **then**: error code/text 009/Cancelation   + **Else** 010/VALIDATIONS FAIL * **If** condition is 14 (DCLEMBOS\_FILE\_DETAIL.CONDITION = 14), and reject reason not C5 (DCLEMBOS\_FILE\_DETAIL.REJECT\_REASON <> 'C5') **then** write data to file2 (for renewals) * Update CRDT.EMBOS\_FILE\_DETAIL (by calling UCDEFD2) |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | KCCT.TRACK\_INFO KCCT.CARDS\_PIN CRDT.EMBOS\_FILE\_DETAIL CRDT.EMBOS\_FILE\_MASTER |

#### [K21DP] Read file that will be sent to LYKOS, Produce file with unique mailer code (to update WAY4)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | K21DPIN1: File with all cards that will be sent to LYKOS  DCL 1 I\_REC,  2 KINDCNT PIC '(06)9' INIT(0),  2 HOLDER CHAR (026) INIT(' '),  2 COMPANY CHAR (026) INIT(' '),  2 TRACK1\_CARD CHAR (016) INIT(' '),  2 STREET CHAR (025) INIT(' '),  2 ZIPCODE CHAR (005) INIT(' '),  2 CITY CHAR (010) INIT(' '),  2 FILE\_EMB\_NAM CHAR (003) INIT(' '),  2 FILE\_EMB\_CNT PIC '99999' INIT(' '),  2 PLS\_DEL\_TYPE CHAR (001) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | K21DPOU1: File containing all cards that are to be sent to LYKOS the respective day along with their unique mailer code  DCL 1 RK21DPOU1,  2 CARD\_NO CHAR(16) INIT(' '),  2 FILE CHAR(15) INIT(' '),  2 ORIG\_REC\_ID CHAR(06) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | READ\_K21DPIN1: More lines in input file |
| **SERVICES** | N/A |
| **Main Process Logic** | For every line of input file: Write data to output file |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every record of input file:   * Write data to output file (NOTE: RK21DPOU1.FILE = I\_REC.FILE\_EMB\_NAM || I\_REC.FILE\_EMB\_CNT) |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

## 

## [Embossing - Dual]

| **Solution Design Details** | |
| --- | --- |
| **Batch Flow Code** | Massive Loading Voucher |
| **Batch Flow Title** | KRPPD013-015 & KRPPD017-019 |
| **Functional Area** | Mainframe z/OS |
| **Source Systems** | Way4 |
| **Target Systems** | Mainframe z/OS |
| **Product** | Debit Cards |

### Functional Description

| **Functional Description** |
| --- |
| * Mainframe z/OS, is responsible for the following tasks: |

### Batch Integration Layer [Mainframe z/OS]

#### 

#### [KRDPD063] Enrich table with data from SYDIPEL (Phase 2-3)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRDPD063** | IEFBR14 | N/A | CRDP.PD.DCPFM.FTOP.KCDDEB01.SEQ,  CRDP.PD.DCPFM.FTOP.KCDCRE01.SEQ | Allocation of working files. |
|  | K51DP | N/A | N/A | Collect customer’s data from SYDIPEL. In case of insufficient data, table is updated with the corresponding reject code and description. Otherwise records are updated with condition 1. |
|  | K52DP | N/A | N/A | Check for debit and credit to have the same condition |
|  | K53DP | CRDP.PD.DCPFM.FTOP.KCDDEB01.SEQ,  CRDP.PD.DCPFM.FTOP.KCDCRE01.SEQ | N/A | Dual Embossing Phase 3:  Select all dual requests from CRDT.EMBOS\_FILE\_DETAIL, and split files per 230000 records |
|  | K54DP | CRDP.PD.DCPFM.FTOP.KCDDEB01.SEQ | CRDP.PD.DCPFM.SEMT.KCDDEB02.SEQ | Produce file with issuing branch (for debit) |
|  | K55DP | CRDP.PD.DCPFM.FTOP.KCDDEB01.SEQ,  CRDP.PD.DCPFM.FTOP.KCDCRE01.SEQ | N/A | Based on files ot be sent to PFM, update records of CRDT.EMBO\_FILE\_DETAIL with condition 2 |
|  | SORT | CRDP.PD.DCPFM.SEMT.KCDDEB02.SEQ | CRDP.PD.DCPFM.SEMT.KCDDEB03.SEQ | Sort by issuing branch |
|  | K56DP | CRDP.PD.DCPFM.SEMT.KCDDEB03.SEQ | CRDP.PD.DCPFM.CMOD.KCDDEB03.SEQ | Production of CMOD report.  Based on the file that we send to PFM for debit, we produce a CMOD file for branches for the following requests:  NEW: New Issuances  PLA: Plastic Replacement  This CMOD is generated only for dual debit cards |
|  | IRXJCL | N/A | N/A | Report issuance-reissuance dual cards.  Run: RUNCMOD CRDP.PD.DCPFM.CMOD.KCDDEB03.SEQ CRD071DPL CDLPL |
|  | IDCAMS | CRDP.PD.DCPFM.FTOP.KCDDEB01.SEQ | CRDP.PD.DCPFM.GDGT.KCCDEB01.BACK(+1) | Debit PFM file backup |
|  | IDCAMS | CRDP.PD.DCPFM.FTOP.KCDCRE01.SEQ | CRDP.PD.DCPFM.GDGT.KCCCRE01.BACK(+1) | Credit PFM file backup |
|  | IEFBR14 | CRDP.PD.DCPFM.SEMT.KCDDEB02.SEQ,  CRDP.PD.DCPFM.SEMT.KCDDEB03.SEQ | N/A | Deallocation of working files |

#### 

#### [KRDPD064] Retrieve files form PFM (DUAL Debit Requests, DUAL Credit Requests) (Phase 4)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRDPD064** | K57DP | CRDP.PD.DCPFM.FTIP.KCDDEB01.SEQ,  CRDP.PD.DCPFM.FTIP.KCDCRE01.SEQ | N/A | Process files retrieved from PFM.  Check tables, validate, and update tables. |
|  | K58DP | CRDP.PD.DCPFM.FTIP.KCDDEB01.SEQ,  CRDP.PD.DCPFM.FTIP.KCDCRE01.SEQ | N/A | Process files retrieved from PFM.  Check and update tables. |
|  | K59DP | N/A | N/A | Check for multiple requests in CRDT.EMBOS\_FILE\_DETAIL with condition 1.  This means that initial file to PFM has more requests than 230000 so we need to collect them and send to PFM (KAMPDECB job)) |
|  | K60DP | N/A | CRDP.PD.DUAL.SEMT.CM60POUT.SEQ | Select from KCCT.SMARRTPIN all requests for SMARTPIN reissuance via mail.  Update corresponding tables with details that matrix has been delivered. CMOD production for requests regarding SMARTPIN reissuance via mail |
|  | SORT | CRDP.PD.DUAL.SEMT.CM60POUT.SEQ | CRDP.PD.DUALC.PERP.CM60POUT.SEQ | Sort by country and postal code |
|  | IDCAMS | CRDP.PD.DCPFM.FTIP.KCDDEB01.SEQ | CRDP.PD.DCPFM.GDGT.KCDDEB01.BACK(+1) | Create PFM Debit file backup |
|  | IDCAMS | CRDP.PD.DCPFM.FTIP.KCDCRE01.SEQ | CRDP.PD.DCPFM.GDGT.KCDCRE01.BACK(+1) | Create PFM Credit file backup |
|  | IEFBR14 | CRDP.PD.DUAL.SEMT.CM60POUT.SEQ | N/A | Deallocation of working files |

#### 

#### [KRDPD065] Process all requests form SMARTPIN reissuance (matrix delivery) via email

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRDPD065** | K82DP | N/A | N/A | Select from KCCT.SMARRTPIN table all requests form SMARTPIN reissuance via email.  Update corresponding tables with details that matrix has been delivered. |

#### 

#### [KRDPD066] Dual Embossing (Phase 5-6)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRDPD066** | IEFBR14 | N/A | CRDP.PD.CPREM.FTOP.CRD06603.SEQ,  CRDP.PD.CPREM.FTOP.CRD06604.SEQ,  CRDP.PD.CPREM.FTOP.CRD08404.SEQ,  CRDP.PD.CPREM.FTOP.CRD08405.SEQ | Allocation of working files |
|  | K83DP | N/A | CRDP.PD.CPREM.SEMT.CRD06601.SEQ,  CRDP.PD.CPREM.SEMT.CRD06602.SEQ | Update requests with condition 4 to 5, and write to file only those with NEW or MMIG for the update of PVV of those cards in WAY4 |
|  | SORT | CRDP.PD.CPREM.SEMT.CRD06601.SEQ | CRDP.PD.CPREM.FTOP.CRD06603.SEQ | Enrich Debit file with header and trailer |
|  | SORT | CRDP.PD.CPREM.SEMT.CRD06602.SEQ | CRDP.PD.CPREM.FTOP.CRD06604.SEQ | Enrich Credit file with header and trailer |
|  | SORT | CRDP.PD.DCPFM.FTIP.KCDDEB01.SEQ | CRDP.PD.DCPFM.SEMT.DDEBPIN.SEQ | Keep only PIN requests from PFM response file (debit) |
|  | SORT | CRDP.PD.DCPFM.FTIP.KCDCRE01.SEQ | CRDP.PD.DCPFM.SEMT.DCREPIN.SEQ | Keep only PIN requests from PFM response file (credit) |
|  | SORT | CRDP.PD.DCPFM.SEMT.DDEBPIN.SEQ,  CRDP.PD.DCPFM.SEMT.DCREPIN.SEQ | CRDP.PD.CPREM.SEMT.CRD08401.SEQ | SOrt Debit and Credit files and keep card and error code |
|  | K84DP | CRDP.PD.CPREM.SEMT.CRD08401.SEQ | CRDP.PD.CPREM.SEMT.CRD08402.SEQ,  CRDP.PD.CPREM.SEMT.CRD08403.SEQ | Retrieve data from CRDT.EMBOS\_FILE\_DETAIL for PIN reissuances, with condition 4.  Create Debit/Credit files. |
|  | SORT | CRDP.PD.CPREM.SEMT.CRD08402.SEQ | CRDP.PD.CPREM.FTOP.CRD08404.SEQ | Enrich Debit file with header and trailer |
|  | SORT | CRDP.PD.CPREM.SEMT.CRD08403.SEQ | CRDP.PD.CPREM.FTOP.CRD08405.SEQ | Enrich Credit file with header and trailer |
|  | IDCAMS | CRDP.PD.CPREM.FTOP.CRD06603.SEQ | CRDP.PD.CPREM.GDGT.CRD06603.BACK(+1) | Create PVV update backup file (Debit) |
|  | IDCAMS | CRDP.PD.CPREM.FTOP.CRD06604.SEQ | CRDP.PD.CPREM.GDGT.CRD06604.BACK(+1) | Create PVV update backup file (Credit) |
|  | IDCAMS | CRDP.PD.CPREM.FTOP.CRD08404.SEQ | CRDP.PD.CPREM.GDGT.CRD08404.BACK(+1) | Create PIN response file backup (Debit) |
|  | IDCAMS | CRDP.PD.CPREM.FTOP.CRD08405.SEQ | CRDP.PD.CPREM.GDGT.CRD08405.BACK(+1) | Create PIN response file backup (Credit) |
|  | IEFBR14 | CRDP.PD.CPREM.SEMT.CRD06601.SEQ,  CRDP.PD.CPREM.SEMT.CRD06602.SEQ,  CRDP.PD.CPREM.SEMT.CRD08401.SEQ,  CRDP.PD.CPREM.SEMT.CRD08402.SEQ,  CRDP.PD.CPREM.SEMT.CRD08403.SEQ,  CRDP.PD.DCPFM.SEMT.DDEBPIN.SEQ,  CRDP.PD.DCPFM.SEMT.DCREPIN.SEQ, | N/A | Deallocation of working files |
|  | CLRSEQ | CRDP.PD.DCPFM.FTIP.KCDDEB01.SEQ,  CRDP.PD.DCPFM.FTIP.KCDCRE01.SEQ | N/A | Clear FTIP |

#### 

#### [KRDPD067] Dual Embossing (Phase 7)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |

| **Mainframe Process** | | | | |
| --- | --- | --- | --- | --- |
| **Job** | **Type** | **Input Files** | **Output Files** | **Description** |
| **KRDPD067** | IEFBR14 | N/A | CRDP.PD.DUALY.FTOP.CRD06703.SEQ,  CRDP.PD.DUALY.FTOP.CRD06706.SEQ,  CRDP.PD.DUALY.FTOP.CRD06707.SEQ | Allocation of working files |
|  | K85DP | N/A | CRDP.PD.DUALY.SEMT.CRD06701.SEQ,  CRDP.PD.DUALY.CMOD.CRD06702.SEQ,  CRDP.PD.DUALY.SEMT.CRD06709.SEQ | Prepare embossing file for SSIS.  Create CMOD for all records which should not be inserted in embossing files |
|  | SORT | CRDP.PD.DUALY.SEMT.CRD06701.SEQ | CRDP.PD.DUALY.FTOP.CRD06703.SEQ | Enrich file with header and trailer |
|  | K86DP | CRDP.PD.DUALY.SEMT.CRD06709.SEQ | N/A | Insert into KCCT.TRACK\_INFO for all records that will be sent to LYKOS personalization bureau |
|  | K87DP | CRDP.PD.DUALY.SEMT.CRD06709.SEQ | N/A | Update field condition to 6 for all these cards that were forwarded to LYKOS. |
|  | K88DP | N/A | CRDP.PD.DUALY.CMOD.CRD06704.SEQ | Create CMOD file for all cards with condition 8 or 9 (rejected records). |
|  | K89DP | N/A | CRDP.PD.DUALY.CMOD.CRD06705.SEQ | Create CMOD for cards sent to LYKOS (group by product code and kind). |
|  | K90DP | N/A | CRDP.PD.DUALY.SEMT.CRD06702.SEQ,  CRDP.PD.KANON.FTOP.CRD06710.SEQ | Collect rejected cards during embossing outgoing process. |
|  | SORT | CRDP.PD.DUALY.SEMT.CRD06702.SEQ | CRDP.PD.DUALY.FTOP.CRD06706.SEQ | Enrich file with header and trailer and copy to FTP file. |
|  | K91DP | CRDP.PD.DUALY.SEMT.CRD06709.SEQ | CRDP.PD.DUALY.SEMT.CRD06703.SEQ | Produce file with unique mailer code for WAY4 update |
|  | SORT | CRDP.PD.DUALY.SEMT.CRD06703.SEQ | CRDP.PD.DUALY.FTOP.CRD06707.SEQ | Enrich file with header and trailer. |
|  | IDCAMS | CRDP.PD.DUALY.FTOP.CRD06703.SEQ | CRDP.PD.DUALY.GDGT.CRD06703.BACK(+1) | Create LYKOS embossing file backup. |
|  | IDCAMS | CRDP.PD.DUALY.FTOP.CRD06706.SEQ | CRDP.PD.DUALY.GDGT.CRD06706.BACK(+1) | Create LYKOS reject file backup. |
|  | IDCAMS | CRDP.PD.DUALY.FTOP.CRD06707.SEQ | CRDP.PD.DUALY.GDGT.CRD06707.BACK(+1) | Create mailer code file backup. |
|  | IEFBR14 | CRDP.PD.DUALY.SEMT.CRD06701.SEQ,  CRDP.PD.DUALY.SEMT.CRD06702.SEQ,  CRDP.PD.DUALY.SEMT.CRD06703.SEQ,  CRDP.PD.DUALY.SEMT.CRD06709.SEQ | N/A | Deallocation of warking files. |
|  | IRXJCL | N/A | N/A | SEBT CMOD file  Run: RUNCMOD CRDP.PD.DUALY.CMOD.CRD06702.SEQ CRD072DPL CDLPL |
|  | IRXJCL | N/A | N/A | SEBT CMOD file  Run: RUNCMOD CRDP.PD.DUALY.CMOD.CRD06704.SEQ CRD073DPL CDLPL |
|  | IRXJCL | N/A | N/A | SEBT CMOD file  Run: RUNCMOD CRDP.PD.DUALY.CMOD.CRD06705.SEQ CRD074DPL CDLPL |

#### [K51DP] Retrieve customer data from SYDIPEL and update tables

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | SRVIXAS: Retrieve card holder’s SYDIPEL (active relation)  SRVIIPS: Retrieve card holder’s basic infromation by SYDIPEL  SRVILOC: Retrieve classification address  SRVIADR: Retrieve address  SRVINMF: Retrieve names  COUSEL2: Retrieve branch’s SYDIPEL  SRVINDS: Retrieve customer’s indicators by SYDIPEL |
| **Main Process Logic** | Retrieve current date.  Select details from CRDT.EMBOS\_FILE\_DETAIL for kind NEW/PLA/PIN/MIG/RWL, product code C06/C67, condition 0, SMARTPIN indicator, and class product DL/CL.  For every records from query: Perform validations, and update tables accordingly. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of query:   * Set FOUND\_OK flag to 0 (all ok) * Check card status:   + **For** kind (DCLEMBOS\_FILE\_DETAIL.KIND) NEW/MIG and   card status (DCLEMBOS\_FILE\_DETAIL.CARD\_STATUS) non-zero or  pin production status (DCLEMBOS\_FILE\_DETAIL.PIN\_PROD\_STATUS) not 2 or  plastic production status (DCLEMBOS\_FILE\_DETAIL.PLA\_PROD\_STATUS) not S or  plastic status (DCLEMBOS\_FILE\_DETAIL.PLA\_STATUS) not I:   * + - Set FOUND\_OK flag to 1     - Update CRDT.EMBOS\_FILE\_DETAIL with rejection code 01 (condition and record status 9, reject reason B1, txt “ΚΑΤΑΡΓΗΘΗΚΕ ΤΟ ΑΙΤΗΜΑ”)   + **For** kind PIN and   card status (DCLEMBOS\_FILE\_DETAIL.CARD\_STATUS) non-zero or  pin production status (DCLEMBOS\_FILE\_DETAIL.PIN\_PROD\_STATUS) not 2:   * + - Set FOUND\_OK flag to 1     - Update CRDT.EMBOS\_FILE\_DETAIL with rejection code 01 (condition and record status 9, reject reason B1, txt “ΚΑΤΑΡΓΗΘΗΚΕ ΤΟ ΑΙΤΗΜΑ”)     - Remove SMARTPIN request from KCCT.SMARTPIN for rejected card:   Select card details with FLG0, and for SQL code 100, try again for FLG 4.  Update KCCT.SMARTPIN with FLG 9, insert data to KCCT.SMARTPIN\_HIST, and then delete it from KCCT.SMARTPIN.   * + **For** kind (DCLEMBOS\_FILE\_DETAIL.KIND) PLA/RWL and   card status (DCLEMBOS\_FILE\_DETAIL.CARD\_STATUS) non-zero or  plastic production status (DCLEMBOS\_FILE\_DETAIL.PLA\_PROD\_STATUS) not S or  plastic status (DCLEMBOS\_FILE\_DETAIL.PLA\_STATUS) not I:   * + - Set FOUND\_OK flag to 1     - Update CRDT.EMBOS\_FILE\_DETAIL with rejection code 01 (condition and record status 9, reject reason B1, txt “ΚΑΤΑΡΓΗΘΗΚΕ ΤΟ ΑΙΤΗΜΑ”) * **If** card was not rejected form above (FOUND\_OK=0) **then**:   + **For** production event (DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT) INS:     - Retrieve branch SYDIPEL from CRAT.C\_OU (by calling COUSEL2)     - For Debit (DCLEMBOS\_FILE\_DETAIL.PRODUCT\_GROUP = '01') or Credit (DCLEMBOS\_FILE\_DETAIL.PRODUCT\_GROUP = '06'), set personal type (DCLEMBOS\_FILE\_DETAIL.PERSONAL\_TYPE) to ‘ΦΠ’ (individual)   + **Otherwise**:     - Set FOUND\_OK flag to 1 (not all ok)     - Retrieve SYDIPEL by card number (by calling SRVIXAS)     - If SYDIPEL was found, then:       * Retrieve address classification (by calling SRVILOC with SYSTEM='15100005' for Debit, and '15100007' for Credit)       * Retrieve first and last name for individual (by calling SRVINMF with I\_CLSF\_IP\_NM\_K=900001)     - For kind NEW/PLA/MIG/RWL:       * **If** SYDIPEL was found (FOUND\_I\_IP=1) and   Individual was found (FOUND\_PERSON\_TYPE=1) and  Address was found (FOUND\_ADDRESS=1) and  Names where found (FOUND\_NAMES=1) and  Combo card field (DCLEMBOS\_FILE\_DETAIL.COMBO\_CARD) is empty **then**:   * + - * + Set FOUND\_OK to 0 (all ok)     - For kind PIN:       * **If** SYDIPEL was found (FOUND\_I\_IP=1) and   Individual was found (FOUND\_PERSON\_TYPE=1) and  Names where found (FOUND\_NAMES=1) and  Combo card field (DCLEMBOS\_FILE\_DETAIL.COMBO\_CARD) is empty **then**:   * + - * + Set FOUND\_OK to 0 (all ok)     - Otherwise: error     - Check if onboarding is pending (by calling SRVINDS)       * **If** client has not accepted onboarding terms (@SRVINDS.ODATA.I\_CLSF\_INDC\_VAL\_K = 8300143) **then**:         + Set FOUND\_OK to 1 (not ok)     - For renewal (DCLEMBOS\_FILE\_DETAIL.KIND = 'RWL'):       * Check if customer is blocked (by calling SRVINDS with @SRVINDS.IDATA.I\_CLSF\_INDC\_TP\_K = 8800007, check for @SRVINDS.ODATA.I\_CLSF\_INDC\_VAL\_K = 8900022)       * Check if customer is under investigation (by calling SRVINDS with @SRVINDS.IDATA.I\_CLSF\_INDC\_TP\_K = 8800007, check for @SRVINDS.ODATA.I\_CLSF\_INDC\_VAL\_K = 8900024)       * Check if transaction is prohibited (by calling SRVINDS with @SRVINDS.IDATA.I\_CLSF\_INDC\_TP\_K = 8200031, check for @SRVINDS.ODATA.I\_CLSF\_INDC\_VAL\_K = 8300001)       * Check is customer has passed away (by calling SRVIIPS, check for @SRVIIPS.ODATA.I\_CLSF\_LGL\_CPCTY\_K = 6600020)       * For any of the above cases, set appropriate flag, and FOUND\_OK flag to 1 (not ok)     - **If** something has gone wrong from the validations (FOUND\_OK=1) **then**:       * **If** customer has not been found **then**: Update CRDT.EMBOS\_FILE\_DETAIL with:         + REJECT\_REASON ='B2'         + REJECT\_REASON\_TXT='ΔΕΝ ΥΦΙΣΤΑΤΑΙ ΣΥ.ΔΙ.ΠΕΛ. ΠΕΛΑΤΗ'         + CONDITION=8         + REC\_STATUS=8       * **Else If** person type has not been found **then**: Update CRDT.EMBOS\_FILE\_DETAIL with:         + REJECT\_REASON ='B3'         + REJECT\_REASON\_TXT='ΔΕΝ ΥΦΙΣΤΑΤΑΙ ΕΓΚΥΡΟΣ ΤΥΠΟΣ ΠΕΛΑΤΗ'         + CONDITION=8         + REC\_STATUS=8       * **ELse If** customer address has not been found **then**: Update CRDT.EMBOS\_FILE\_DETAIL with:         + REJECT\_REASON ='B4'         + REJECT\_REASON\_TXT='ΔΕΝ ΥΦΙΣΤΑΤΑΙ ΔΙΕΥΘΥΝΣΗ ΠΕΛΑΤΗ'         + CONDITION=8         + REC\_STATUS=8       * **Else If** customer first and last name have not been found **then**: Update CRDT.EMBOS\_FILE\_DETAIL with:         + REJECT\_REASON ='B5'         + REJECT\_REASON\_TXT='ΔΕΝ ΥΦΙΣΤΑΤΑΙ ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΠΕΛΑΤΗ'         + CONDITION=8         + REC\_STATUS=8       * **Else If** onboarding is pending for customer **then**: Update CRDT.EMBOS\_FILE\_DETAIL with:         + REJECT\_REASON ='B6'         + REJECT\_REASON\_TXT='ONBOARDING ΠΕΛΑΤΗΣ ΔΕΝ ΑΠΟΔΕΧΘΗΚΕ ΟΡΟΥΣ'         + CONDITION=8         + REC\_STATUS=8       * **Else If** combo card field is empty **then**: Update CRDT.EMBOS\_FILE\_DETAIL with:         + REJECT\_REASON ='B7'         + REJECT\_REASON\_TXT='ΤΟ ΠΕΔΙΟ COMBO CARD ΕΙΝΑΙ ΚΕΝΟ'         + CONDITION=10         + REC\_STATUS=10       * **Else If** customer is blocked **then**: Update CRDT.EMBOS\_FILE\_DETAIL with:         + REJECT\_REASON ='C2'         + REJECT\_REASON\_TXT='ΠΕΛΑΤΗΣ BLOCKED'         + CONDITION=14         + REC\_STATUS=14       * **Else If** transaction is prohibited **then**: Update CRDT.EMBOS\_FILE\_DETAIL with:         + REJECT\_REASON ='C3'         + REJECT\_REASON\_TXT='ΠΕΛΑΤΗΣ ΜΕ ΑΠΑΓΟΡΕΥΣΗ ΚΙΝΗΣΗΣ'         + CONDITION=14         + REC\_STATUS=14       * **Else If** customer has passed away **then**: Update CRDT.EMBOS\_FILE\_DETAIL with:         + REJECT\_REASON ='C4'         + REJECT\_REASON\_TXT='ΠΕΛΑΤΗΣ ΘΑΝΟΝΤΑΣ'         + CONDITION=14         + REC\_STATUS=14       * **Else If** customer is under investigation **then**: Update CRDT.EMBOS\_FILE\_DETAIL with:         + REJECT\_REASON ='C5'         + REJECT\_REASON\_TXT='ΠΕΛΑΤΗΣ UNDERINV'         + CONDITION=14         + REC\_STATUS=14     - **If** embossing name field (DCLEMBOS\_FILE\_DETAIL.PE\_EMBOSSING\_NAME) is not empty, **then**:       * Retrieve I\_IP\_ET\_KAT from card number (by calling SRVIXAS with @SRVIXAS.IDATA.I\_CLSF\_ACC\_X\_TP\_K = 15000018) * **If** card was not rejected form above (FOUND\_OK=0) **then**:   + Update CRDT.EMBOS\_FILE\_DETAIL with (all ok):     - REJECT\_REASON =’ ‘     - REJECT\_REASON\_TXT=' '     - CONDITION=1     - REC\_STATUS=1 * (Note: For any rejected card, we have already marked it with condition 8 (check card again next time) or condition 9 (reject request)) |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER  KCCT.SMARTPIN  KCCT.SMARTPIN\_HIST |

#### [K52DP] Validate Debit and Credit agree on condition

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve current date.  Select debit details (D\_CARD,D\_COMBO\_CARD,D\_CONDITION,D\_REC\_STATUS,D\_REJ\_REASON,D\_REJ\_REASON\_TXT) from CRDT.EMBOS\_FILE\_DETAIL for king MIG/NEW/PLA/RWL/PIN, condition 1/8/9, product code C06, class product DL, and SMARTPIN indicator.  For every records from query: Retrieve Credit requests. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of query:   * Retrieve credit requests from CRDT.EMBOS\_FILE\_DETAIL for kind MIG/NEW/PLA/RWL/PIN, condition 1,8,9 (C\_CONDITION), product code C67, class product CL, and SMARTPIN indicator. * **If** credit record has not been found (SQL code 100) **then**:   + Update CRDT.EMBOS\_FILE\_DETAIL with condition and record status 10, reject reason ‘B9’ and text ‘ΔΕΝ ΒΡΕΘΗΚΕ CREDIT DUAL ΕΓΓΡΑΦΗ’ * **Else**:   + **If** credit condition (C\_CONDITION) is 1, and debit condition (D\_CONDITION) is 8/9 **then**:     - Update CRDT.EMBOS\_FILE\_DETAIL with debit condition, records status, rejection status, and reason text   + **If** debit condition (D\_CONDITION) is 1 and credit condition (C\_CONDITION) is 8/9 **then**:     - Update CRDT.EMBOS\_FILE\_DETAIL with credit condition, records status, rejection status, and reason text |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### [K53DP] Dual Embossing: Retrieve Requests from table (Phase 3)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | K53DPOU1: Debit file  DCL 1 RFTODPMS,  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME CHAR(30) INIT(' '),  2 EXP\_DATE CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '),  2 ACTION\_CODE CHAR(01) INIT(' '),  2 MAILER\_CODE CHAR(01) INIT(' '),  2 IDENTITY CHAR(15) INIT(' '),  2 PIN\_BRANCH CHAR(05) INIT(' '),  2 TYP\_REC CHAR(03) INIT(' '),  2 E\_PIN CHAR(01) INIT(' '),  2 FILLER CHAR(03) INIT(' '),  2 COMBO\_CARD CHAR(19) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | K53DPOU2: Credit file  DCL 1 RFTODPMS,  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME CHAR(30) INIT(' '),  2 EXP\_DATE CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '),  2 ACTION\_CODE CHAR(01) INIT(' '),  2 MAILER\_CODE CHAR(01) INIT(' '),  2 IDENTITY CHAR(15) INIT(' '),  2 PIN\_BRANCH CHAR(05) INIT(' '),  2 TYP\_REC CHAR(03) INIT(' '),  2 E\_PIN CHAR(01) INIT(' '),  2 FILLER CHAR(03) INIT(' '),  2 COMBO\_CARD CHAR(19) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve current date.  Select details from CRDT.EMBOS\_FILE\_DETAIL for kind NEW/PLA/PIN/MIG/RWL, product code C06/C67, condition 1, SMARTPIN indicator, and class product DL/CL.  For every records from query: Set values for output file, and write data to output file 1 for debit, and to output file 2 for credit. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of query:   * Set FOUND\_OK flag to 0 (all ok) * **If** all is ok (FOUND\_OK = 0) **then**:   + Set values from debit and credit files for PFM:     - For production event (DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT) INS, set customer name (RFTODPMS.CUSTNAME) to ‘MASTERCARD CARDHOLDER’     - For kind NEW/PIN/MIG:       * Set RFTODPMS.MAILER\_CODE to ‘N’       * Set RFTODPMS.E\_PIN to ‘Y’     - For kind PLA/RWL:       * Set RFTODPMS.MAILER\_CODE to ‘N’       * Set RFTODPMS.E\_PIN to ‘N’     - For production event () INS: Set RFTODPMS.INDENTITY to ‘CRA ‘     - Otherwise: Set RFTODPMS.INDENTITY to ‘CRA ‘||DCLEMBOS\_FILE\_DETAIL.I\_IP   + **If** product code (DCLEMBOS\_FILE\_DETAIL.PRODUCT\_CODE) is ‘C06’ **then**:     - Write data to output file 1   + **Else**:     - Write data to output file 2 |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER |

#### 

#### [K54DP] Produce file with issuing branch (Debit)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | INPFILE1: File contains new requests for embossing with PIN delivery branch  DCL 1 I\_RFTODPMS,  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME CHAR(30) INIT(' '),  2 EXP\_DATE CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '),  2 ACTION\_CODE CHAR(01) INIT(' '),  2 MAILER\_CODE CHAR(01) INIT(' '),  2 IDENTITY CHAR(15) INIT(' '),  2 PIN\_BRANCH CHAR(05) INIT(' '),  2 TYP\_REC CHAR(03) INIT(' '),  2 E\_PIN CHAR(01) INIT(' '),  2 FILLER CHAR(03) INIT(' '),  2 COMBO\_CARD CHAR(19) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | OUTFILE1: File contains new requests for embossing with issuing branch  DCL 1 O\_RFTODPMS,  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME CHAR(30) INIT(' '),  2 EXP\_DATE CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '),  2 ACTION\_CODE CHAR(01) INIT(' '),  2 MAILER\_CODE CHAR(01) INIT(' '),  2 IDENTITY CHAR(15) INIT(' '),  2 ISSUING\_BRANCH CHAR(05) INIT(' '),  2 TYP\_REC CHAR(03) INIT(' '),  2 E\_PIN CHAR(01) INIT(' '),  2 FILLER CHAR(03) INIT(' '),  2 COMBO\_CARD CHAR(19) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE: More data in input file |
| **SERVICES** | SCDEFD2: Retrieve details from CRDT.EMBOS\_FILE\_DETAIL |
| **Main Process Logic** | For every records from input file: Retrieve details for card, and write data to output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * Retrieve details for card (with condition 1) from CRDT.EMBOS\_FILE\_DETAIL (by calling SCDEFD2) * Write data to output file (Note: add 2 spaces in front of branch value) |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### 

#### [K55DP] Update records in table with condition 2

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | K55DPIN: Debit/Credit file  DCL 1 RK55DPIN,  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME CHAR(30) INIT(' '),  2 EXP\_DATE CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '),  2 ACTION\_CODE CHAR(01) INIT(' '),  2 MAILER\_CODE CHAR(01) INIT(' '),  2 IDENTITY CHAR(15) INIT(' '),  2 PIN\_BRANCH CHAR(05) INIT(' '),  2 TYP\_REC CHAR(03) INIT(' '),  2 E\_PIN CHAR(01) INIT(' '),  2 FILLER CHAR(03) INIT(' '),  2 COMBO\_CARD CHAR(19) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_K55DPIN: More data in input file |
| **SERVICES** | SCDTRC: Retrieve latest card record from KCCT.TRACK\_INFO  ICDTRC: Insert records into KCCT.TRACK\_INFO |
| **Main Process Logic** | Retrieve current date.  For every records from input file: Check for error cases, retrieve records, and write to output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * **If** type (RK55DPIN.TYP\_REC) is not NEW/PLA/MIG/RWL/PIN, **then** error * Retrieve latest record with card details from CRDT.EMBOS\_FILE\_DETAIL with condition 1 * **If** condition (DCLEMBOS\_FILE\_DETAIL.CONDITION) is 0, **then** error * **Else if** condition is greater than 2, **then** error * **Else if** condition is 2, **then** program has already run, continue on (no error) * **Else** update card record in CRDT.EMBOS\_FILE\_DETAIL with condition/record status 2 from condition 1 * Retrieve latest track of card from KCCT.TRACK\_INFO (by calling SCDTRC) * Insert record into KCCT.TRACK\_INFO (by calling ICDTRC) |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER |

#### 

#### [K56DP] Production of CMOD report for branches

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | K56DPIN:  DCL 1 RK56DPIN,  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME CHAR(30) INIT(' '),  2 EXP\_DATE CHAR(04) INIT(' '),  2 SVC CHAR(03) INIT(' '),  2 ACTION\_CODE CHAR(01) INIT(' '),  2 FILLER CHAR(01) INIT(' '),  2 IDENTITY CHAR(15) INIT(' '),  2 PIN\_BRANCH0 CHAR(02) INIT(' '),  2 PIN\_BRANCH CHAR(03) INIT(' '),  2 TYP\_REC CHAR(03) INIT(' '),  2 E\_PIN CHAR(01) INIT(' '),  2 FILLER2 CHAR(03) INIT(' '),  2 COMBO\_CARD CHAR(19) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | K56DPOUT:  DCL FLINE CHAR(133) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_K56DPIN: More data in input file |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve current date.  For every records from input file: Retrieve card records from table, and write data to output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * For type (RK56DPIN.TYP\_REC) NEW/PLA:   + Select card records from CRDT.EMBOS\_FILE\_DETAIL with condition 2   + **If** local variable branch value (WS\_BRANCH) doesn’t agree with PIN branch from input file (RK56DPIN.PIN\_BRANCH) **then**:     - Update local variable branch value (WS\_BRANCH)     - Initialize variables for page/serial number (WS\_CNTREAD\_BRANCH) for branch   + Write data to output file.     - (Note: For production event (DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT) PLA/RWL/RSU set local variable WS\_CATEGORY to ‘ ΕΠΑΝΕΚΔΟΣΗ’ and for FRD/FCC/INS/LST/MIG/NEW/EXP/STL/MAS to ‘ ΝΕΑ ΕΚΔΟΣΗ’)     - FLINE=' '!!WS\_CNTREAD\_BRANCH!!(02)' '!!   SUBSTR(RK56DPIN.CARD,1,4)!!'\*\*\*\*\*\*'!!  SUBSTR(RK56DPIN.CARD,11,6)!!(02)' '!!  RK56DPIN.CUSTNAME!!(04)' '!!  SUBSTR(RK56DPIN.IDENTITY,5,11)!!(03)' '!!  WS\_CATEGORY!!(02)' '!!  DCLEMBOS\_FILE\_DETAIL.PRODUCT\_NAME; |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER |

#### 

#### [K57DP] Process files from PFm, Validate data and update tables

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | K57DPIN:  DCL 1 RK57DPIN UNALIGNED BASED(ADDR(RRK57DPIN)),  2 CARD CHAR(16) INIT(' '),  2 CUSTNAME CHAR(30) INIT(' '),  2 PVV CHAR(04) INIT(' '),  2 CVV2 PIC'(03)9' INIT(0),  2 CVV1 PIC'(03)9' INIT(0),  2 ICVV PIC'(03)9' INIT(0),  2 PIN\_BLOCK CHAR(16) INIT(' '),  2 ERROR\_CDE CHAR(02) INIT(' '),  2 TYP\_REC CHAR(03) INIT(' '),  2 E\_PIN CHAR(01) INIT(' '),  2 MATRIX CHAR(15) INIT(' '),  2 FILLER CHAR(03) INIT(' '),  2 COMBO\_CARD CHAR(19) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_K57DPIN: More data in input file |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve current date.  Retrieve records from CRDT.EMBOS\_FILE\_DETAIL with kind NEW/PLA/PIN/MIG/RWL with condition 3, SMARTPIN indicator, product code C06/C67, and class product DL/CL. Update record in CRDT.EMBOS\_FILE\_DETAIL to condition and record status 2.  For every records from input file: Retrieve card record details from table, perform validations, and update record condition in table. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * Retrieve (embossing data) last card record from CRDT.EMBOS\_FILE\_DETAIL with product code C06/C67, and class product DL/CL * **If** file record’s code (RK57DPIN.ERROR\_CDE) is non-zero, **then** error * **If** file record’s condition (DCLEMBOS\_FILE\_DETAIL.CONDITION) is 3, **then** error * **If** file record’s condition is not 2, **then** error * **If** file record type (RK57DPIN.TYP\_REC) does not agree with kind from table (DCLEMBOS\_FILE\_DETAIL.KIND), **then** error * For file record type not NEW/PIN/PLA/MIG/RWL, error * Update record in CRDT.EMBOS\_FILE\_DETAIL to condition and record status 3. |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  ~~CRDT.EMBOS\_FILE\_MASTER~~ |

#### 

#### [K58DP] Process files from PFm, Validate data and update tables

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | K58DPIN:  DCL 1 RK58DPIN,  2 CARD CHAR (16) INIT(' '),  2 CUSTNAME CHAR (30) INIT(' '),  2 PVV CHAR (04) INIT(' '),  2 CVV2 PIC '(03)9' INIT(0),  2 CVV1 PIC '(03)9' INIT(0),  2 ICVV PIC '(03)9' INIT(0),  2 PIN\_BLOCK CHAR (16) INIT(' '),  2 ERROR\_CDE CHAR (02) INIT(' '),  2 TYP\_REC CHAR (03) INIT(' '),  2 E\_PIN CHAR (01) INIT(' '),  2 MATRIX CHAR (15) INIT(' '),  2 FILLER CHAR(03) INIT(' '),  2 COMBO\_CARD CHAR(19) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_K58DPIN: More data in input file |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve current date.  Retrieve count of records from table CRDT.EMBOS\_FILE\_DETAIL with kind NEW/PLA/PIN/MIG/RWL, condition 2. SMARTPIN indicator, product code C06, C67, and class product DL/CL. **If** incomplete requests were found (records were retrieved) **then** error.  For every records from input file: Retrieve information from tables, update tables, and perform validations. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * Retrieve card details (last record) from CRDT.EMBOS\_FILE\_DETAIL with condition 3, product code C06/C67, class product DL/CL * For type (RK58DPIN.TYP\_REC) PLS/RWL no action * Otherwise, retrieve latest card track from CRDT.SMARTPIN\_TRACK * **If** condition (DCLEMBOS\_FILE\_DETAIL.CONDITION) is 3, **then**:   + Update card (RK58DPIN.CARD) record for specified file ID (DCLEMBOS\_FILE\_DETAIL.FILE\_ID) in CRDT.EMBOS\_FILE\_DETAIL to condition/record status 4, and PVV/CVV1/CVV2/ICVV/PRIN\_BLOCK from input file values   + For type (RK58DPIN.TYP\_REC) NEW/PIN/MIG:     - If type is PIN and DCLSMARTPIN\_TRACK.MATRIX\_SEND\_TYPE is 0/1/2 then:       * Retrieve latest card record from KCCT.SMARTPIN with condition 3       * Update card record in KCCT.SMARTPIN with FLG 0, to FLG 2, and update matrix (with input file value) and update timestamp     - Else:       * Retrieve latest card record from KCCT.SMARTPIN       * Update card matrix in KCCT.SMARTPIN based in INS\_TMSTMP     - Update card record in CRDT.SMARTPIN\_TRACK with MATRIX\_SEND\_STATUS, MATRIX\_START\_DATE, MATRIX\_END\_DATE, MATRIX\_SEND\_DATE, TIMESTAMP   + For type PLA/RWL, no action   + Otherwise, error * **Else if** condition is less than 2, **then**: error * **Else if** condition is greater than 4, **then**: error |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER  CRDT.SMARTPIN\_TRACK  KCCT.SMARTPIN |

#### 

#### [K59DP] Check for multiple requests with condition 1 for card

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_K59DPIN: More data in input file |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve current date.  Retrieve count of records from table CRDT.EMBOS\_FILE\_DETAIL with kind NEW/PLA/PIN/RWL, condition 1. SMARTPIN indicator, product code C06, C67, and class product DL/CL.  If records are returned (incomplete) or SQL code 100/-305 gets returned, error. Process must be rerun. |
| **PROGRAM MAIN LOOP** | N/A |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### 

#### [K60DP] Retrieve SMARTPIN reissuance by mail requests, Update tables, Create file

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | K60DPOUT:  DCL 1 RK60DPOUT,  2 CARD CHAR (16) INIT(' '),  2 FIL01 CHAR (01) INIT(' '),  2 REQUEST CHAR (01) INIT(' '),  2 FIL02 CHAR (01) INIT(' '),  2 REQ\_DATE CHAR (10) INIT(' '),  2 FIL03 CHAR (01) INIT(' '),  2 SND\_DATE CHAR (10) INIT(' '),  2 FIL04 CHAR (01) INIT(' '),  2 HOLDER CHAR (26) INIT(' '),  2 FIL05 CHAR (01) INIT(' '),  2 CUST\_NAME CHAR (40) INIT(' '),  2 FIL06 CHAR (01) INIT(' '),  2 COMP\_NAME CHAR (40) INIT(' '),  2 FIL07 CHAR (01) INIT(' '),  2 CRA\_ADDR CHAR (40) INIT(' '),  2 FIL08 CHAR (01) INIT(' '),  2 CRA\_TK CHAR (40) INIT(' '),  2 FIL09 CHAR (01) INIT(' '),  2 CRA\_CITY CHAR (40) INIT(' '),  2 FIL10 CHAR (01) INIT(' '),  2 CRA\_COUNTRY CHAR (40) INIT(' '),  2 FIL11 CHAR (01) INIT(' '),  2 MATRIX CHAR (15) INIT(' '),  2 FIL12 CHAR (01) INIT(' '),  2 FLAG\_ENT CHAR (02) INIT('NO'),  2 FIL13 CHAR (01) INIT(' '),  2 FLAG\_LET CHAR (02) INIT('NO'),  2 FIL14 CHAR (01) INIT(' '),  2 CARD\_ORIGIN CHAR (02) INIT(' '),  2 FIL15 CHAR (01) INIT(' '),  2 PRODUCT CHAR (09) INIT(' '),  2 FIL16 CHAR (01) INIT(' '),  2 PRODUCT\_NAME CHAR (30) INIT(' '),  2 FIL17 CHAR (01) INIT(' '),  2 CNT PIC '(6)9' INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | SRVINMF: Retrieve names from CRAT.C\_IP\_X\_NM  SRVIXAL: Check account-customer relation  UCDSPT2: Update SEND\_STATUS, SEND\_TYPE & SEND\_DATE into CRDT.SMARTPIN\_TRACK  SMRPIN: Update MATRIX value into KCCT.SMARTPIN |
| **Main Process Logic** | Retrieve current date.  Retrieve records from KCCT.SMARTPIN with card origin DC/CC, request 1, FLG\_DELIVER 0/1, MATRIX not empty, FLG 2, and product 000000C06,000000C67, and card information from KCCT.SMARTPIN/CRDT.EMBOS\_FILE\_DETAIL with card origin DC/CC, request 3, FLG\_DELIVER 0/1, MATRIX not empty, FLG 2, condition 4, and product code C06/C67.  For every records from query: Update tables, and if possible, retrieve customer data, and write to output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of query:   * Update FLG to 1, and MATRIX to masked version (mask digits 4-14) in table KCCT.SMARTPIN with CARD, INS\_TMSTMP, CARD\_ORIGIN,REQUEST, FLG\_DELIVER,MATRIX, FLG values from input file * Update table CRDT.SMARTPIN\_TRACK (SEND\_TYPE/SEND\_STATUS to 1, and MATRIX\_SEND\_DATE) (by calling UCDSPT2) * **If** delivery type is not via branch (RKCCSPIN.FLG\_DELIVER ^= '0') **then**:   + Retrieve customer data (by calling SRVIXAL)     - Note:   For RKCCSPIN.PRODUCT = '000000C06',  @SRVIXAL.IDATA.I\_CLSF\_IT\_ACC\_K = 15100005 (Debit)  Else @SRVIXAL.IDATA.I\_CLSF\_IT\_ACC\_K = 15100007 (Credit)  Set @SRVIXAL.IDATA.I\_CLSF\_ACC\_X\_TP\_K = 15000001 (Holder)   * + Retrieve customer first and last name (by calling SRVINMF with @SRVINMF.IDATA.I\_CLSF\_IP\_NM\_K = 900001)   + For Debit (RKCCSPIN.PRODUCT = '000000C06'), write data to output file:     - For request (RKCCSPIN.REQUEST) 3 (PIN reissuance):       * Category ‘ ΝΕΑ ΕΚΔΟΣΗ’       * Retrieve latest card record from CRDT.EMBOS\_FILE\_DETAIL with condition 4     - For request 1 (PIN reminder):       * Category ' REMIND PIN' |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | KCCT.SMARTPIN  CRDT.SMARTPIN\_TRACK  CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER |

#### 

#### [K82DP] Process Requests from SMARTPIN reissuance (matrix delivery) via email

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | CROSMS: Send SMS (Windows-Local) |
| **Main Process Logic** | Retrieve current date.  Retrieve records from KCCT.SMARTPIN and CRDT.EMBOS\_FILE\_DETAIL with card origin DC/CC, request 3, FLG\_DELIVERY 2 (email), FLG 2 (inform form Q & PFM),MATRIX non-empty, PRODUCT\_CODE C06/C67, and CLASS\_PRODUCT DL/CL.  For every records from query: Send email, and update tables. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of query:   * For Debit (RKCCSPIN.PRODUCT = '000000C06'), make savepoint, and send email (bu calling CROSMS) ( * **If** email wasn’t sent successfully, **then** rollback to savepoint * **Else**:   + Update card record FLG to 1 (completed) in KCCT.SMARTPIN for product '000000C06'/'000000C67'   + Retrieve latest card record from CRDT.SMARTPIN\_TRACK   + Update card record MATRIX\_SEND\_STATUS to 1 and MATRX\_SEND\_DATE in CRDT.SMARTPIN\_TRACK   + Commit |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | KCCT.SMARTPIN  CRDT.SMARTPIN\_TRACK  CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER |

#### 

#### [K83DP] Update conditions from 4 to 5, and create files

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | K83DPOU1: File containing card and PVV (Debit)  DCL 1 O\_PINRESPONSE,  2 O\_CARD CHAR(19) INIT(' '),  2 O\_PVV CHAR(04) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | K83DPOU2: File containing card and PVV (Credit)  DCL 1 O\_PINRESPONSE,  2 O\_CARD CHAR(19) INIT(' '),  2 O\_PVV CHAR(04) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | UCDEFD1: Update into CRDT.EMBOS\_FILE\_DETAIL |
| **Main Process Logic** | Retrieve records from CRDT.EMBOS\_FILE\_DETAIL with kind PIN, class product DL/CL, product code C06/C67, condition 4, and smartpin indicator.  For every records from query: Write data to output files, and update table. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of query:   * For kind (DCLEMBOS\_FILE\_DETAIL.KIND) NEW/MIG   + Write data to output files:     - For product code (DCLEMBOS\_FILE\_DETAIL.PRODUCT\_CODE) C06, write data to output file 1     - Otherwise, write data to output file 2 * Update into CRDT.EMBOS\_FILE\_DETAIL (by calling UCDEFD1) |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### 

#### [K84DP] Retrieve card information, and create files (Debit/Credit)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | K84DPIN1:  DCL 1 RK84DPIN1 ,  2 CARD CHAR (019) INIT(' '),  2 ERROR\_CDE CHAR (002) INIT(' '),  2 KND CHAR (003) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | K84DPOU1: For Debit  DCL 1 PINRESPONSE,  2 O\_WAY4FILENAME CHAR (040) INIT(' '),  2 O\_TRANSACTION\_ID CHAR (255) INIT(' '),  2 O\_BATCH\_ID CHAR (032) INIT(' '),  2 O\_PVV CHAR (004) INIT(' '),  2 O\_ERROR\_CDE CHAR (002) INIT(' '),  2 O\_CARD CHAR (019) INIT(' '),  2 O\_COMBO\_CARD CHAR (019) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | K84DPOU2: For Credit  DCL 1 PINRESPONSE,  2 O\_WAY4FILENAME CHAR (040) INIT(' '),  2 O\_TRANSACTION\_ID CHAR (255) INIT(' '),  2 O\_BATCH\_ID CHAR (032) INIT(' '),  2 O\_PVV CHAR (004) INIT(' '),  2 O\_ERROR\_CDE CHAR (002) INIT(' '),  2 O\_CARD CHAR (019) INIT(' '),  2 O\_COMBO\_CARD CHAR (019) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | L\_MORE\_K84DPIN1: More data in input file |
| **SERVICES** | UCDEFM1: Update CRDT.EMBOS\_FILE\_MASTER FILE\_STATUS  UCDEFD1: Mask PVV in CRDT.EMBOS\_FILE\_DETAIL  UCDEFD3: Update CRDT.EMBOS\_FILE\_MASTER condition from 5 to 6, and PVV  SCDTRC: Select from KCCT.TRACK\_INFO  ICDTRC: Insert into KCCT.TRACK\_INFO  SCDEFD3: Select from CRDT.EMBOS\_FILE\_MASTER and CRDT.EMBOS\_FILE\_DETAIL |
| **Main Process Logic** | Retrieve current date.  For every records from input file: Retrieve card information, write data to output files, and update tables. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of input file:   * Retrieve card information from CRDT.EMBOS\_FILE\_MASTER (by calling SCDEFD3) * Write data to output files:   + For Debit (@SCDEFD3.ODATA.PRODUCT\_CODED = 'C06') write to output file 1   + Otherwise write to output file 2 * Update CRDT.EMBOS\_FILE\_MASTER (by calling UCDEFM1) * Update CRDT.EMBOS\_FILE\_DETAIL (by calling UCDEFD1) * Retrieve latest record from KCCT.TRACK\_INFO (by calling SCDTRC) * Insert details into KCCT.TRACK\_INFO (by calling ICDTRC) |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER |

#### [K85DP] Prepare files, embossing and CMOD

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | K85DPOU1: Embossing file for SSIS (Dual)  DCL 1 REMBOS,  2 KINDCNT PIC '(06)9' INIT(0),  2 CC01 CHAR (001) INIT('{'),  2 CC02 CHAR (003) INIT('DCC'),  2 CC03 CHAR (001) INIT('{'),  2 CC04 CHAR (001) INIT('{'),  2 CC05 CHAR (003) INIT('EMB'),  2 CC06 CHAR (001) INIT('{'),  2 LINE1\_CARD CHAR (019) INIT(' '),  2 CC07 CHAR (001) INIT('}'),  2 PLASTIC\_COLOR CHAR (001) INIT(' '),  2 FILLER01 CHAR (011) INIT(' '),  2 MM\_LHX CHAR (002) INIT(' '),  2 CC08 CHAR (001) INIT('/'),  2 YY\_LHX CHAR (002) INIT(' '),  2 FILLER02 CHAR (010) INIT(' '),  2 CC09 CHAR (001) INIT('}'),  2 HOLDER CHAR (026) INIT(' '),  2 CC10 CHAR (001) INIT('}'),  2 COMPANY CHAR (026) INIT(' '),  2 CC11 CHAR (001) INIT('{'),  2 CC12 CHAR (003) INIT('OCR'),  2 CC13 CHAR (001) INIT('{'),  2 FILLER03 CHAR (012) INIT(' '),  2 CARD\_4LAST PIC '9999' ,  2 FILLER04 CHAR (001) INIT(' '),  2 CVV2\_INDENT PIC '999' ,  2 FILLER05 CHAR (001) INIT(' '),  2 CC14 CHAR (001) INIT('{'),  2 CC15 CHAR (003) INIT('ENC'),  2 CC16 CHAR (003) INIT('{%B'),  2 TRACK1\_CARD CHAR (016) INIT(' '),  2 CC17 CHAR (001) INIT('^'),  2 TRACK1\_NAME CHAR (026) ,  2 CC18 CHAR (001) INIT('^'),  2 YYMM\_LHXH\_1 PIC '9999' ,  2 SVC\_1 CHAR (003) INIT(' '),  2 CVV1\_1 PIC '999' ,  2 ZEROES\_10 CHAR (010) INIT('0000000000'),  2 CC19 CHAR (001) INIT('?'),  2 FILLER06 CHAR (001) INIT(' '),  2 CC20 CHAR (001) INIT(';'),  2 TRACK2\_CARD CHAR (016) INIT(' '),  2 CC21 CHAR (001) INIT('='),  2 YYMM\_LHXH\_2 PIC '9999' ,  2 SVC\_2 CHAR (003) INIT(' '),  2 CVV1\_2 PIC '999' ,  2 CARD\_SEQ\_NUM PIC '99' ,  2 ZEROES\_8 CHAR (008) INIT('00000000'),  2 CC22 CHAR (001) INIT('?'),  2 FILLER07 CHAR (001) INIT(' '),  2 CC23 CHAR (002) INIT('|1'),  2 CC24 CHAR (002) INIT('\_0'),  2 CC25 CHAR (011) INIT('[4037 0883]'),  2 TEMPL\_NAME CHAR (012) INIT(' '),  2 CC26 CHAR (003) INIT('[1]'),  2 KINDCNT2 CHAR (006) ,  2 CC27 CHAR (003) INIT('[2]'),  2 BRANCH CHAR (003) ,  2 CC28 CHAR (003) INIT('[3]'),  2 SNAME CHAR (022) ,  2 CC29 CHAR (003) INIT('[4]'),  2 PNAME CHAR (001) ,  2 CC30 CHAR (003) INIT('[5]'),  2 FNAME CHAR (003) ,  2 CC31 CHAR (003) INIT('[6]'),  2 STREET CHAR (025) ,  2 CC32 CHAR (003) INIT('[7]'),  2 ZIPCODE CHAR (005) ,  2 CC33 CHAR (003) INIT('[8]'),  2 CITY CHAR (010) ,  2 CC34 CHAR (003) INIT('[9]'),  2 HMER CHAR (010) INIT(' '),  2 CC35 CHAR (003) INIT('[A]'),  2 POS\_LIMIT CHAR (005) ,  2 CC36 CHAR (003) INIT('[B]'),  2 WDRW\_LIMIT CHAR (005) ,  2 CC37 CHAR (003) INIT('[C]'),  2 FILE\_EMB\_NAM CHAR (003) INIT(' '),  2 FILE\_EMB\_CNT PIC '99999' ,  2 CC38 CHAR (003) INIT('[D]'),  2 MATRIX CHAR (015) INIT(' '),  2 CC39 CHAR (003) INIT('[K]'),  2 INS\_AR\_CARD CHAR (016) INIT(' '),  2 CC40 CHAR (003) INIT('[L]'),  2 INS\_AR\_DATE CHAR (010) INIT(' '),  2 CC41 CHAR (003) INIT('[M]'),  2 INS\_AR\_NAME CHAR (027) INIT(' '),  2 CC42 CHAR (003) INIT('[N]'),  2 INT\_LIMIT CHAR (005) ,  2 CC43 CHAR (003) INIT('[R]'),  2 DOM\_CNP\_LIMIT CHAR (005) ,  2 CC44 CHAR (003) INIT('[S]'),  2 INT\_CNP\_LIMIT CHAR (005) ,  2 CC45 CHAR (003) INIT('[T]'),  2 DOM\_WTH\_LIMIT CHAR (005) ,  2 CC46 CHAR (003) INIT('[U]'),  2 INT\_WTH\_LIMIT CHAR (005) ,  2 ICVV\_USER PIC '999' ,  2 PIN\_BLOCK CHAR (016) INIT(' '),  2 FILLER08 CHAR (001) INIT(' '),  2 PLS\_DEL\_TYPE CHAR (001) INIT(' '),  2 COMBO\_CARD CHAR (016) INIT(' '),  2 C\_SVC CHAR (003) INIT(' '),  2 C\_CVV2 PIC '999' ,  2 C\_ICVV PIC '999' ,  2 CC47 CHAR (003) INIT('[F]'),  2 CREDIT\_LIMIT CHAR(07) INIT(' '),  2 FILLER09 CHAR(01) INIT(' '),  2 EURO\_01 CHAR(04) INIT('ΕΥΡΩ'),  2 FILLER10 CHAR(48) INIT(' '),  2 CC48 CHAR (003) INIT('[G]'),  2 C\_WDRW\_LIMIT CHAR (005) ,  2 FILLER11 CHAR (035) INIT(' '),  2 CC49 CHAR (003) INIT('{F{'),  2 CC50 CHAR (003) INIT('@@@'); |
| **OUTFILE2 FILE DECLARATION** | CMODFILE: CMOD file with exceptions  DCL LINE CHAR (133) INIT(' '); |
| **OUTFILE3 FILE DECLARATION** | K85DPOU2:  DCL 1 REMBOALL,  2 KINDCNT PIC '(06)9' INIT(0),  2 HOLDER CHAR (026) INIT(' '),  2 COMPANY CHAR (026) INIT(' '),  2 TRACK1\_CARD CHAR (016) INIT(' '),  2 STREET CHAR (025) INIT(' '),  2 ZIPCODE CHAR (005) INIT(' '),  2 CITY CHAR (010) INIT(' '),  2 FILE\_EMB\_NAM CHAR (003) INIT(' '),  2 FILE\_EMB\_CNT PIC '99999' INIT(' '),  2 PLS\_DEL\_TYPE CHAR (001) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | UCDEFD3: Update CRDT.EMBOS\_FILE\_MASTER and CRDT.EMBOS\_FILE\_DETAIL  SCDEPM: Select from CRDT.EMBOS\_PARM  SCDECR: Select from CRDT.EMBOS\_COUNTER  UCDECR: Update into CRDT.EMBOS\_COUNTER  SCCSPN1: Select from KCCT.SMARTPIN  UCCSPN1: Update into KCCT.SMARTPIN  UCDEFM2: Update into CRDT.EMBOS\_FILE\_MASTER |
| **Main Process Logic** | Retrieve current date.  Retrieve records from CRDT.EMBOS\_FILE\_DETAIL and CRDT.EMBOS\_FILE\_MASTER (by FILE\_ID) with kind NEW/PLA/MIG/RWL, condition 5, product code C06 and class product DL.  For every records from query: Perform Validations, write data to output file, and update tables.  If LYKOS\_STATUS\_CNT is greater than 0, then update CRDT.EMBOS\_FILE\_MASTER (by calling UCDEFM2), for Debit. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of query:   * Retrieve combo card details from CRDT.EMBOS\_FILE\_DETAIL and CRDT.EMBOS\_FILE\_MASTER (by FILE\_ID) with condition 5, product code C67 and class product CL. * For anything but plastic replacements requests   ((DCLEMBOS\_FILE\_DETAIL.KIND ^= 'PLA') &  (DCLEMBOS\_FILE\_DETAIL.KIND ^= 'RWL') &  (DCLEMBOS\_FILE\_DETAIL.SMARTPIN = 'Y')):   * + Retrieve details from KCCT.SMARTPIN (by calling SCCSPN1)   + Mask matrix in KCCT.SMARTPIN (bu calling UCCSPN1)   + The above 2 are run for both Debit and Credit * **If** previous WAY4 file name does not agree with current file name, **then**:   + If LYKOS\_STATUS\_CNT is greater than 0 then:     - Update CRDT.EMBOS\_FILE\_MASTER (by calling UCDEFM2) for Debit   + Initialize counters (including LYKOS\_STATUS\_CNT)   + Retrieve file name from CRDT.EMBOS\_PARM (by calling SCDEPM)   + Retrieve last counter from CRDT.EMBOS\_COUNTER (by calling SCDECR)   + Update last counter used in CRDT.EMBOS\_COUNTER (by calling UCDECR) * For renewal (DCLEMBOS\_FILE\_DETAIL.KIND = 'RWL'):   + Retrieve file name form CRDT.EMBOS\_PARM (by calling SCDEPM) * Increment LYKOS\_STATUS\_CNT * For Mastercard (DCLEMBOS\_FILE\_DETAIL.SCHEME=’MC’)   + Format output (Note: If production event is not instant (DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT != 'INS') then format differently) * Otherwise error, scheme not recognised * Validate:   + Embossing name: DCLEMBOS\_FILE\_DETAIL.PRODUCT\_GROUP=’01’(Dual Debit) and DCLEMBOS\_FILE\_DETAIL.EMBOSSING\_NAME is non empty, otherwise error   + Expiration date: non-empty, in format DD.MM.YYYY, with valid day, month, and year not before 2023, otherwise error   + SVC: DCLEMBOS\_FILE\_DETAIL.PRODUCT\_CODE = C06, and DCLEMBOS\_FILE\_DETAIL.SVC not 226, otherwise error   + SNAME: For any production event(DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT) other than Insert (INS), DCLEMBOS\_FILE\_DETAIL.SNAME can’t be empty   + Street: For any production event(DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT) other than Insert (INS), DCLEMBOS\_FILE\_DETAIL.STREET can’t be empty   + Plastic Delivery Type: Only valid delivery type values (DCLEMBOS\_FILE\_DETAIL.PL\_DELIVERY\_TYPE) are 02/04/07/09   + PIN Block: DCLEMBOS\_FILE\_DETAIL.PIN\_BLOCK can’t be empty   + Cardholder Card Number: Card number (DCLEMBOS\_FILE\_DETAIL.CARD) can only include digits 0-9 and spaces, and none of the 4 card number quadruplets can be empty   + Branch: For any production event(DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT) other than Insert (INS), 3 first characters of pin delivery branch number (DCLEMBOS\_FILE\_DETAIL.PIN\_DELIVERY\_BRANCH) can’t be all empty   + Insertion Date: For any production event(DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT) other than Insert (INS), DCLEMBOS\_FILE\_DETAIL.ISUUING\_DATE can’t be all empty   + File Embossing Name: For any production event(DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT) other than Insert (INS), L\_EMBOS\_PARM.LYKOS\_PREFIX can’t be all empty   + File Cardholder Name, Track1 name and company: For any production event(DCLEMBOS\_FILE\_DETAIL.PRODUCTION\_EVENT) other than Insert (INS), for Dual Debit (DCLEMBOS\_FILE\_DETAIL.PRODUCT\_GROUP = ‘01’) holder name and tarck1 can include A-Z, 0-9, spaces, and special characters, otherwise error   + In any of the above cases, if an error get’s found, the rest of the validations are bypassed * **If** no error found, **then**:   + Evaluate record id (Note: WS\_TYPE = DCLEMBOS\_FILE\_MASTER.PRODUCTION\_CODE || DCLEMBOS\_FILE\_MASTER.PRODUCT\_CODE). For valid values, increase counter, and set it to REMBOS.KINDCNT,REMBOS.KINDCNT2:     - New issuance-Dual Debit Mastercard ('AC06','IC06','MC06')     - Plastic Replacement-Dual Debit Mastercard ('BC06')     - Migration-Dual Debit Mastercard ('CC06')     - Renewal-Dual Debit Mastercard ('EC06')     - For any other value, error   + For Mastercard (DCLEMBOS\_FILE\_DETAIL.SCHEME=’MC’), write data to output file 1 (embossing file to sent to LYKOS), for other values, error   + Write data to output file 3 (K85DPOU2) based on DCLEMBOS\_FILE\_DETAIL.CARD   + Write data to output file 3 (K85DPOU2) based on COM\_CARD\_NO * **Else** there’s been a validation error:   + Write data to output file 2 (CMODFILE) * Update CRDT.EMBOS\_FILE\_DETAIL sensitive data with encryption characters (by calling UCDEFD3), for Debit and for Credit |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  CRDT.EMBOS\_FILE\_MASTER  DXPT.GEOST |

#### [K86DP] Insert records to table to be sent to LYKOS for personalization bureau

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | K86DPI: Embossing file containing all records that will be sent to LYKOS  DCL 1 REMBOS,  2 KINDCNT PIC '(06)9' INIT(0),  2 HOLDER CHAR (026) INIT(' '),  2 COMPANY CHAR (026) INIT(' '),  2 TRACK1\_CARD CHAR (016) INIT(' '),  2 STREET CHAR (025) INIT(' '),  2 ZIPCODE CHAR (005) INIT(' '),  2 CITY CHAR (010) INIT(' '),  2 FILE\_EMB\_NAM CHAR (003) INIT(' '),  2 FILE\_EMB\_CNT PIC '99999' INIT(' '),  2 PLS\_DEL\_TYPE CHAR (001) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | READ\_K86DPI: More data from input file |
| **SERVICES** | ICDTRC: Insert into KCCT.TRACK\_INFO  SCDEFD2: Select details from CRDT.EMBOS\_FILE\_DETAIL  SCDTRC: Select card details from KCCT.TRACK\_INFO |
| **Main Process Logic** | Retrieve current date.  For every records from input file: Retrieve details from table, and insert new details. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of query:   * Retrieve record details from CRDT.EMBOS\_FILE\_DETAIL (by calling SCDEFD2) * Retrieve details from KCCT.TRACK\_INFO (by calling SCDTRC) * Insert information into KCCT.TRACK\_INFO (by calling ICDTRC) and commit every 100 inserts |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | KCCT.TRACK\_INFO  CRDT.EMBOS\_FILE\_DETAIL |

#### [K87DP] Update condition to 6 for all cards forwarded to LYKOS

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | K87DPIN:  DCL 1 RK87DPIN,  2 FILLER1 CHAR(058) INIT(' '),  2 CARD CHAR(016) INIT(' '),  2 FILLER2 CHAR(049) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | N/A |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_K87DPIN: More data from input file |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve current date.  For every records from input file: Retrieve latest card details form table, perform validations, and if needed update card records. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of query:   * Retrieve latest card details from CRDT.EMBOS\_FILE\_DETAIL * **If** it is an anonymous card (DCLEMBOS\_FILE\_DETAIL.CONDITION < 5), error * **Else if** it is a rejected card (DCLEMBOS\_FILE\_DETAIL.CONDITION > 6), error * **Else if** process has run again (DCLEMBOS\_FILE\_DETAIL.CONDITION = 6), continue for further information * **ELse**: Update card record condition and record status to 6 and update UPD\_TMSTMP in CRDT.EMBOS\_FILE\_DETAIL * Commit |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### [K88DP]Create CMOD file for all cards with condition 8 or 9 (rejected)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | K88DPOUT:  DCL FLINE CHAR(133) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve current date.  Select records from CRDT.EMBOS\_FILE\_DETAIL with kind NEW/PLA/PIN/MIG/RWL, condition 8/9 (rejected), product code C06, and class product DL.  For every records from input file: Write data to output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of query:   * **If** local saved branch doesn’t agree with issuing branch form table (DCLEMBOS\_FILE\_DETAIL.ISSUING\_BRANCH) **then**:   + Update local branch variable, page initialization, and serial number (WS\_CNTREAD\_BRANCH) for branch * Write data to output file:   + '1 '!!'ΕΘΝΙΚΗ ΤΡΑΠΕΖΑ ΤΗΣ ΕΛΛΑΔΟΣ Α.Ε'!!(59)' '!!'ΚΩΔΙΚΟΣ ΚΑΤΑΣΤΑΣΗΣ: CRD073-DPL';   + ' '!!'ΠΡΟΣ ΚΑΤΑΣΤΗΜΑ: '!!WS\_BRANCH!!' - '   !!WS\_BRANCH\_TITLE  !!(41)' '!!'ΗΜΕΡΑ: '!!WS\_DATE\_CUR  !!' '!!'(ΩΡΑ: '!!WS\_TIME\_CUR!!')';   * + (3)' '!!'ΚΑΤΑΣΤΑΣΗ '   !!'ΑΝΕΚΠΛΗΡΩΤΩΝ ΑΙΤΗΜΑΤΩΝ ΕΚΔΟΣΕΩΝ/'  !!'ΕΠΑΝΕΚΔΟΣΕΩΝ COMBO DUAL ΚΑΡΤΩΝ'  !!(45)' '!!'ΣΕΛΙΣ '!!ANALPAGE;   * + ' '!!(03)' '!!'Α/Α'   !!(07)' '!!'ΚΑΡΤΑ'  !!(15)' '!!'ΟΝΟΜΑΤΕΠΩΝΥΜΟ'  !!(13)' '!!'ΣΥ.ΔΙ.ΠΕΛ.'  !!(06)' '!!'ΑΙΤΗΜΑ'  !!(08)' '!!'ΚΩΔΙΚΟΣ / AΙΤΙΟΛΟΓΙΑ ΑΠΟΡΡΙΨΗΣ';   * + FLINE=WS\_CNTREAD\_BRANCH!!(5)' '!!   SUBSTR(DCLEMBOS\_FILE\_DETAIL.CARD,1,4)!!'\*\*\*\*\*\*'!!  SUBSTR(DCLEMBOS\_FILE\_DETAIL.CARD,11,6)!!' '!!  FSTR\_CUSTNAME!!' '!!  DCLEMBOS\_FILE\_DETAIL.I\_IP!!(01)' '!!  WS\_CATEGORY!!(01)' '!!  DCLEMBOS\_FILE\_DETAIL.REJECT\_REASON!!  (01)' '!!SUBSTR(REJECT\_REASON\_TXT,1,40);   * + Note:   FSTR\_CUSTNAME=CLEAR(DCLEMBOS\_FILE\_DETAIL.SNAME)!!' '!!  CLEAR(DCLEMBOS\_FILE\_DETAIL.FNAME)!!' '!!  CLEAR(DCLEMBOS\_FILE\_DETAIL.PNAME);   * + Note: DCLEMBOS\_FILE\_DETAIL.KIND='NEW': WS\_CATEGORY=' ΝΕΑ ΕΚΔΟΣΗ '   DCLEMBOS\_FILE\_DETAIL.KIND='PIN': WS\_CATEGORY=' ΕΠΑΝΕΚΔΟΣΗ PIN '  DCLEMBOS\_FILE\_DETAIL.KIND='PLA': WS\_CATEGORY='ΕΠΑΝ/ΣΗ ΠΛΑΣΤΙΚΟΥ '  DCLEMBOS\_FILE\_DETAIL.KIND='MIG': WS\_CATEGORY=' MIGRATION '  DCLEMBOS\_FILE\_DETAIL.KIND='RWL': WS\_CATEGORY='ΕΠΑΝ/ΣΗ ΛΟΓΩ ΛΗΞΗΣ' |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL  DXPT.GEOST |

#### [K89DP] Create CMOD for cards sent to LYKOS (group by product code, kind)

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | OUTFILE1: CMOD  DCL ULINE CHAR (133) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | MORE\_INPUT1: More data in input file |
| **SERVICES** | N/A |
| **Main Process Logic** | Retrieve current date.  Select records (count:, issuing branch) from CRDT.EMBOS\_FILE\_DETAIL with kind MIG/NEW/PLA/RWL, condition 6, product code C06/C67, and class product DL/CL (grouped by issuing branch).  For every records from input file: Retrieve card information from table, perform validations, and write data to output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of query:   * Retrieve branch title from DXPT.GEOST * Select records (product code, kind, count) from CRDT.EMBOS\_FILE\_DETAIL with kind MIG/NEW/PLA/RWL, condition 6, product code C06/C67, class product DL/CL (grouped by product code and kind) * For every retrieved record:   + Evaluate result of fetched count based on kind and product (Note:L\_KDPR = L\_KND !! L\_PRD)     - Valid values with specified counters set to count: MIGC06,MIGC67,NEWC06,NEWC67,PLAC06,PLAC67,RWLC06,RWLC67     - Otherwise set “other” counter * Check total between Debit and Credit for dual (for NEW/MIG/PLA/RWL) * Write data to output file:   + '1 '!!'ΕΘΝΙΚΗ ΤΡΑΠΕΖΑ ΤΗΣ ΕΛΛΑΔΟΣ Α.Ε.'   + ' '!!'ΚΩΔΙΚΟΣ ΚΑΤΑΣΤΑΣΗΣ : CRD074-DPL'   + ' '!!'ΠΡΟΣ KATAΣΤΗΜΑ: '!! WS\_BRANCH!!' - '!!WS\_BRANCH\_TITLE   + ' '!!'ΗΜΕΡΑ: '!!SUBSTR(L\_DATE\_CUR,1,2)!!'.' !!SUBSTR(L\_DATE\_CUR,4,2)!!'.'!!SUBSTR(L\_DATE\_CUR,7,4)   + (3)' '!!'ΚΑΤΑΣΤΑΣΗ ΑΙΤΗΜΑΤΩΝ ΕΚΔΟΣΕΩΝ/ΕΠΑΝΕΚΔΟΣΕΩΝ'   !!' COMBO DUAL ΚΑΡΤΩΝ'  !!(50)' '!!'ΣΕΛΙΣ '!!L\_PAGE;   * + ' '!!(131)'-'   + ' '!!'ΣΥΝΟΛΙΚΟΣ ΑΡΙΘΜΟΣ ΚΑΡΤΩΝ: '!!DEC\_TO\_CHAR(L\_FCNT)   + ' '!!(131)'-'   + ' '!!'ΑΝΑΛΥΤΙΚΑ: '   + ' '!!(131)'-'   + 'DUAL MASTERCARD CARD(DEBIT)',   'C06',  L\_NEWCNTC06,  L\_MIGCNTC06,  L\_PLACNTC06,  L\_RWLCNTC06   * + ' '!!(131)'-'   + ' ΛΟΙΠΟΙ ΣΥΝΔΥΑΣΜΟΙ ΕΙΔΩΝ ΚΑΙ ΠΡΟΙΟΝΤΩΝ'   + ' '!!(131)'-'   + ' ΛΟΙΠΑ (ΣΥΝΟΛΙΚΑ) : '!!DEC\_TO\_CHAR(L\_OTHCNT)   + ' '!!(131)'-' |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | CRDT.EMBOS\_FILE\_DETAIL |

#### [K90DP] Collect rejected cards during embossing outgoing process

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | N/A |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | K90DP1O:  DCL 1 RK90DPO,  2 O\_REC\_NO CHAR (06) INIT(' '),  2 O\_CARD\_EMB CHAR (26) INIT(' '),  2 O\_CARD\_NO CHAR (16) INIT(' '),  2 O\_FILLER CHAR (02) INIT(' '),  2 O\_REJ\_CODE CHAR (03) INIT(' '),  2 O\_REJ\_DESCR CHAR (100) INIT(' '),  2 O\_FILE CHAR (15) INIT(' '),  2 O\_ELTA\_DATE CHAR (10) INIT(' '),  2 O\_ORIG\_REC\_ID CHAR (06) INIT(' '),  2 O\_EXCEPT\_FLG CHAR (01) INIT(' '),  2 O\_PLS\_DEL\_TYPE CHAR (02) INIT(' '),  2 WAY4\_FILE\_NAME CHAR (40) INIT(' '),  2 TRANSACTION\_ID CHAR (255) INIT(' '),  2 BATCH\_ID CHAR (32) INIT(' '),  2 O\_COMBO\_CARD CHAR (19) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | K90DP2O:  DCL 1 RK90DP2O,  2 CARD CHAR (16) INIT(' '),  2 FILLER\_1 CHAR (01) INIT(';'),  2 SYDIPEL CHAR (11) INIT(' '),  2 FILLER\_2 CHAR (01) INIT(';'),  2 CLASSIFIER\_VALUE CHAR (01) INIT(' '),  2 FILLER\_3 CHAR (01) INIT(';'),  2 CLASSIFIER\_NAME CHAR (16) INIT(' '); |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | N/A |
| **SERVICES** | UCDEFD2: Update CRDT.EMBOS\_FILE\_DETAIL  SRVIXAS: Find SYDIPEL through card number |
| **Main Process Logic** | Retrieve current date.  Select records from CRDT.EMBOS\_FILE\_DETAIL / CRDT.EMBOS\_FILE\_MASTER (by FILE\_ID) with record status 8/9/10/14, condition 8/9/10/14, class product DL and product code C06.  For every records from input file: Retrieve combo card information, and write data to output files for card and combo card. Update table for card and combo card, to condition 11. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of query:   * Retrieve combo card and condition from CRDT.EMBOS\_FILE\_DETAIL for card number with record status and condition 8/9/10/14, class product CL, and product code C67. * Write data to output file 1 (rejections) for card (DCLEMBOS\_FILE\_DETAIL.CARD)   + Note:   For condition (DCLEMBOS\_FILE\_DETAIL.CONDITION) 8 RK90DPO.O\_REJ\_CODE = '008';  RK90DPO.O\_REJ\_DESCR = 'MISSING SIDIPEL DATA';  For condition (DCLEMBOS\_FILE\_DETAIL.CONDITION) 9 RK90DPO.O\_REJ\_CODE = '009';  RK90DPO.O\_REJ\_DESCR = 'CANCELATION';  For condition (DCLEMBOS\_FILE\_DETAIL.CONDITION) 10 RK90DPO.O\_REJ\_CODE = '010';  RK90DPO.O\_REJ\_DESCR = ‘VALIDATIONS FAIL’;   * For condition (DCLEMBOS\_FILE\_DETAIL.CONDITION) 14, and reject reason (DCLEMBOS\_FILE\_DETAIL.REJECT\_REASON) not C5:   + Write data to output file 2 for card (DCLEMBOS\_FILE\_DETAIL.CARD) and combo card. * Update CRDT.EMBOS\_FILE\_DETAIL for card (DCLEMBOS\_FILE\_DETAIL.CARD) (by calling UCDEFD2) with condition (to @UCDEFD2.IDATA.COND\_STATE), and 11 (to CONDITION and REC\_STATUS) * Similarly, if combo card was found, update CRDT.EMBOS\_FILE\_DETAIL for combo card (by calling UCDEFD2) |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | KCCT.TRACK\_INFO  KCCT.CARDS\_PIN  CRDT.EMBOS\_FILE\_DETAIL CRDT.EMBOS\_FILE\_MASTER |

#### [K91DP] Produce file with unique mailer code for WAY3 update

| **IL Process** | |
| --- | --- |
| **Process Name** | IL Mainframe |
| **INPFILE1 FILE DECLARATION** | K91DPI: File that will be sent to LYKOS  DCL 1 I\_REC,  2 KINDCNT PIC '(06)9' INIT(0),  2 HOLDER CHAR (026) INIT(' '),  2 COMPANY CHAR (026) INIT(' '),  2 TRACK1\_CARD CHAR (016) INIT(' '),  2 STREET CHAR (025) INIT(' '),  2 ZIPCODE CHAR (005) INIT(' '),  2 CITY CHAR (010) INIT(' '),  2 FILE\_EMB\_NAM CHAR (003) INIT(' '),  2 FILE\_EMB\_CNT PIC '99999' INIT(' '),  2 PLS\_DEL\_TYPE CHAR (001) INIT(' '); |
| **INPFILE2 FILE DECLARATION** | N/A |
| **OUTFILE1 FILE DECLARATION** | K91DPO: File containing the cards that are to be sent to LYKOS the respective day along with their unique mailer code  DCL 1 RK91DPO,  2 CARD\_NO CHAR(16) INIT(' '),  2 FILE CHAR(15) INIT(' '),  2 ORIG\_REC\_ID CHAR(06) INIT(' '); |
| **OUTFILE2 FILE DECLARATION** | N/A |
| **INTERNAL TABLES/ARRAYS** | N/A |
| **CRITICAL BOOLEAN INDICATORS** | READ\_K91DPI: More data in input file |
| **SERVICES** | N/A |
| **Main Process Logic** | For every records from input file: Write card data to output file. |
| **PROGRAM MAIN LOOP** | Inside the main loop the following processes take place for every line of query:   * Write data to output file for I\_REC.TRACK1\_CARD   + Note: RK91DPO.FILE = I\_REC.FILE\_EMB\_NAM || I\_REC.FILE\_EMB\_CNT   RK91DPO.ORIG\_REC\_ID = I\_REC.KINDCNT |
| **Arrays filled with corresponding values** | N/A |
| **CMS TABLES** | N/A |

#### 

#### 

#### 

#### 

### [CRDT.GL\_ACC\_PARM] – Accounting Data Parametrization

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | CRDT.GL\_ACC\_PARM | | | |  |
| **Table description** | | | Table maintaining all the information related to an organization and its parametrization. | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | BUSINESS\_EVENT | Business event. Either the code of organization to be paid in case that CHANNEL = 'O' Or the Business Event as received from Way4 in case that CHANNEL = 'B' | | Yes | CHAR | 32 | |
| 2 | TRAN\_CDE | Transaction code | |  | CHAR | 3 | |
| 3 | TXN\_CHAN |  | |  | CHAR | 4 | |
| 4 | CHANNEL | Operation Mode 'O': Online 'B': Batch | | Yes | CHAR | 3 | |
| 5 | AMT\_TYPE | Amount type 'F': Fee 'A': Amount | | Yes | CHAR | 1 | |
| 6 | AMT\_TYPE\_SEQ | In case that more than one fees need to be applied for a business event, this column contains their sequence (1,2,3…) | | Yes | INTEGER | 1 | |
| 7 | DESCRIPTION | Business Event description | |  | CHAR | 20 | |
| 8 | GL\_OPERATION\_ID |  | |  | CHAR | 5 | |
| 9 | EFFECTIVE\_DATE | Effective Date | |  | CHAR | 10 | |
| 10 | TIMESTAMP | Timestamp | |  | CHAR | 26 | |
| 11 | ACCOUNT | Account number | |  | CHAR | 16 | |
| 12 | DEBIT\_ACC\_IND | Debit Account Indicator | |  | CHAR | 1 | |
| 13 | SERVICE |  | |  | CHAR | 8 | |
| 14 | BATCH\_G\_SPECIAL |  | |  | CHAR | 3 | |
| 15 | TRT\_REASON |  | |  | CHAR | 80 | |
| 16 | INSTITUTION |  | |  | CHAR | 6 | |
| 17 | PRODUCT | Product code | |  | CHAR | 4 | |
| 18 | TRANS\_CURR |  | |  | CHAR | 3 | |

### [CRDT.SMARTPIN\_TRACK] – SmartPin Data

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **CRDT.SMARTPIN\_TRACK** | | | |  |
| **Table description** | | | Table containing SmartPin data | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | CARD |  | |  | CHAR | 19 | |
| 2 | PRODUCT\_NAME | Product name | |  | CHAR | 30 | |
| 3 | MATRIX\_START\_DATE |  | |  | CHAR | 10 | |
| 4 | MATRIX\_END\_DATE |  | |  | CHAR | 10 | |
| 5 | MATRIX\_SEND\_DATE |  | |  | CHAR | 1 | |
| 6 | MATRIX\_SEND\_STATUS |  | |  | CHAR | 1 | |
| 7 | MATRIX\_SEND\_DATE |  | |  | CHAR | 10 | |
| 8 | POSITIONS\_SEND\_TYPE |  | |  | CHAR | 1 | |
| 9 | POSITIONS\_SEND\_STATUS |  | |  | CHAR | 1 | |
| 10 | POSITIONS\_SEND\_DATE |  | |  | CHAR | 10 | |
| 11 | TIMESTAMP |  | |  | CHAR | 26 | |

### [KCCT.CRCA\_TPAYMENTS] – Logging payments

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **KCCT.CRCA\_TPAYMENTS** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | BRANCH | Branch number | |  | CHAR | 5 | |
| 2 | SEQNO |  | |  | CHAR | 4 | |
| 3 | TERMID |  | |  | CHAR | 4 | |
| 4 | TELLERID |  | |  | CHAR | 5 | |
| 5 | TRAN\_CODE | Transaction code | |  | CHAR | 4 | |
| 6 | TRAN\_DATE | Transaction date | |  | CHAR | 10 | |
| 7 | TRAN\_TIME | Transaction time | |  | CHAR | 8 | |
| 8 | TMSTMP | Timestamp | |  | CHAR | 26 | |
| 9 | INSTITUTION |  | |  | CHAR | 8 | |
| 10 | REFNO |  | |  | CHAR | 40 | |
| 11 | TRAN\_KEY | Transaction key | |  | CHAR | 18 | |
| 12 | AUTHID |  | |  | CHAR | 6 | |
| 13 | CARD |  | |  | CHAR | 16 | |
| 14 | S\_BRANCH | Source Branch | |  | CHAR | 5 | |
| 15 | S\_ACCOUNT | Source Account | |  | CHAR | 16 | |
| 16 | T\_BRANCH | Target Branch | |  | CHAR | 5 | |
| 17 | T\_ACCOUNT | Target Account | |  | CHAR | 16 | |
| 18 | AMOUNT |  | |  | FLOAT | 15 | |
| 19 | COMMISION |  | |  | FLOAT | 15 | |
| 20 | INST\_NUM |  | |  | INTEGER | 2 | |
| 21 | BATCH\_DATE |  | |  | CHAR | 10 | |
| 22 | REV\_BRANCH |  | |  | CHAR | 5 | |
| 23 | REV\_SEQNO |  | |  | CHAR | 4 | |
| 24 | REV\_TERMID |  | |  | CHAR | 4 | |
| 25 | REV\_TELLERID |  | |  | CHAR | 5 | |
| 26 | REV\_TRAN\_CODE |  | |  | CHAR | 4 | |
| 27 | REV\_TRAN\_DATE |  | |  | CHAR | 10 | |
| 28 | REV\_TRAN\_TIME |  | |  | CHAR | 8 | |
| 29 | REV\_TMSTMP |  | |  | CHAR | 26 | |
| 30 | INT\_KEY |  | |  | CHAR | 25 | |

### [CRDT.UFX\_DEBLOCK]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **CRDT.UFX\_DEBLOCK** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | TRX\_TYPE\_CODE |  | |  | CHAR | 32 | |
| 2 | TRX\_DESC |  | |  | CHAR | 80 | |
| 3 | DEBLOCK\_IND |  | |  | CHAR | 1 | |
| 4 | EFFECTIVE\_DATE |  | |  | CHAR | 10 | |
| 5 | CLEARING\_IND |  | |  | CHAR | 1 | |

### [KCCT.CARDS\_PIN]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **KCCT.CARDS\_PIN** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | CARD |  | |  | CHAR | 16 | |
| 2 | STATUS\_EKD |  | |  | CHAR | 1 | |
| 3 | STATUS\_PIN |  | |  | CHAR | 1 | |
| 4 | ACTIVE |  | |  | CHAR | 1 | |
| 5 | SECOND\_EXP\_DATE |  | |  | CHAR | 10 | |
| 6 | HOLDER\_ET\_KAT |  | |  | CHAR | 26 | |
| 7 | MATRIX |  | |  | CHAR | 20 | |
| 8 | MATRIX\_SND\_TYP |  | |  | CHAR | 1 | |
| 9 | MATRIX\_SND\_STAT |  | |  | CHAR | 1 | |
| 10 | MATRIX\_START\_DATE |  | |  | CHAR | 10 | |
| 11 | MATRIX\_END\_DATE |  | |  | CHAR | 10 | |
| 12 | POSITIONS\_SND\_TYP |  | |  | CHAR | 1 | |
| 13 | POSITIONS\_SND\_STAT |  | |  | CHAR | 1 | |
| 14 | MATRIX\_SND\_DATE |  | |  | CHAR | 10 | |
| 15 | POSITIONS\_SND\_DATE |  | |  | CHAR | 10 | |
| 16 | PRODUCT |  | |  | CHAR | 9 | |
| 17 | PLASTIC\_TYPE |  | |  | CHAR | 1 | |
| 18 | PVV\_NUM |  | |  | CHAR | 4 | |
| 19 | PIN\_SYNC |  | |  | CHAR | 4 | |
| 20 | REF\_ACTION |  | |  | CHAR | 1 | |
| 21 | REF\_TMSTMP |  | |  | CHAR | 26 | |
| 22 | PIN\_BRANCH |  | |  | CHAR | 5 | |
| 23 | FLG\_VIP |  | |  | CHAR | 1 | |
| 24 | FLG\_CARD\_DELIVER |  | |  | CHAR | 1 | |
| 25 | FLG\_STMT\_DELIVER |  | |  | CHAR | 1 | |
| 26 | ID |  | |  | CHAR | 20 | |
| 27 | NEW\_CARD |  | |  | CHAR | 16 | |
| 28 | COMMISSION\_DATE |  | |  | CHAR | 10 | |
| 29 | ADDR\_DATE |  | |  | CHAR | 10 | |
| 30 | MICROTAG\_TYPE |  | |  | CHAR | 1 | |
| 31 | MICROTAG\_COLOR |  | |  | CHAR | 1 | |
| 32 | CARD\_PERSONILIZED |  | |  | CHAR | 1 | |
| 33 | REFERENCE |  | |  | CHAR | 20 | |
| 34 | EVENT\_CODE |  | |  | CHAR | 2 | |

### [KCCT.CRCA\_TPAYMENTS]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **KCCT.CRCA\_TPAYMENTS** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | BRANCH |  | |  | CHAR | 5 | |
| 2 | SEQNO |  | |  | CHAR | 4 | |
| 3 | TERMID |  | |  | CHAR | 4 | |
| 4 | TELLERID |  | |  | CHAR | 5 | |
| 5 | TRAN\_CODE |  | |  | CHAR | 4 | |
| 6 | TRAN\_DATE |  | |  | CHAR | 10 | |
| 7 | TRAN\_TIME |  | |  | CHAR | 8 | |
| 8 | TMSTMP |  | |  | CHAR | 26 | |
| 9 | INSTITUTION |  | |  | CHAR | 8 | |
| 10 | REFNO |  | |  | CHAR | 40 | |
| 11 | TRAN\_KEY |  | |  | CHAR | 18 | |
| 12 | AUTHID |  | |  | CHAR | 6 | |
| 13 | CARD |  | |  | CHAR | 16 | |
| 14 | S\_BRANCH |  | |  | CHAR | 5 | |
| 15 | S\_ACCOUNT |  | |  | CHAR | 16 | |
| 16 | T\_BRANCH |  | |  | CHAR | 5 | |
| 17 | T\_ACCOUNT |  | |  | CHAR | 16 | |
| 18 | AMOUNT |  | |  | DEC FIXED | (15,2) | |
| 19 | COMMISSION |  | |  | DEC FIXED | (15,2) | |
| 20 | INST\_NUM |  | |  | DEC FIXED | (2,0) | |
| 21 | BATCH\_DATE |  | |  | CHAR | 10 | |
| 22 | REV\_BRANCH |  | |  | CHAR | 5 | |
| 23 | REV\_SEQNO |  | |  | CHAR | 4 | |
| 24 | REV\_TERMID |  | |  | CHAR | 4 | |
| 25 | REV\_TELLERID |  | |  | CHAR | 5 | |
| 26 | REV\_TRAN\_CODE |  | |  | CHAR | 4 | |
| 27 | REV\_TRAN\_DATE |  | |  | CHAR | 10 | |
| 28 | REV\_TRAN\_TIME |  | |  | CHAR | 8 | |
| 29 | REV\_TMSTMP |  | |  | CHAR | 26 | |
| 30 | INT\_KEY |  | |  | CHAR | 25 | |

### [CRDT.PAYMENT\_ORDER]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **CRDT.PAYMENT\_ORDER** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | FILE\_ID |  | |  | CHAR | 32 | |
| 2 | RECORD\_ID |  | |  | CHAR | 32 | |
| 3 | SRN |  | |  | CHAR | 32 | |
| 4 | ERROR\_MSG |  | |  | CHAR | 100 | |
| 5 | ACCOUNT |  | |  | CHAR | 32 | |
| 6 | DEP\_ACCOUNT |  | |  | CHAR | 16 | |
| 7 | PAYMENT\_AMOUNT |  | |  | DEC FIXED | (15,2) | |
| 8 | PAYED\_AMOUNT |  | |  | DEC FIXED | (15,2) | |
| 9 | PAYMENT\_METHOD |  | |  | CHAR | 10 | |
| 10 | DUE\_DATE |  | |  | CHAR | 10 | |
| 11 | PAYMENT\_STATUS |  | |  | CHAR | 1 | |
| 12 | UPDATE\_TIMESTAMP |  | |  | CHAR | 26 | |
| 13 | INSERT\_TIMESTAMP |  | |  | CHAR | 26 | |
| 14 | PRODUCT\_CODE |  | |  | CHAR | 8 | |

### [CRDT.PAYMENT\_ORDER\_HIS]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **CRDT.PAYMENT\_ORDER\_HIS** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | FILE\_ID |  | |  | CHAR | 32 | |
| 2 | RECORD\_ID |  | |  | CHAR | 32 | |
| 3 | SRN |  | |  | CHAR | 32 | |
| 4 | ERROR\_MSG |  | |  | CHAR | 100 | |
| 5 | ACCOUNT |  | |  | CHAR | 32 | |
| 6 | DEP\_ACCOUNT |  | |  | CHAR | 16 | |
| 7 | PAYMENT\_AMOUNT |  | |  | DEC FIXED | (15,2) | |
| 8 | PAYED\_AMOUNT |  | |  | DEC FIXED | (15,2) | |
| 9 | PAYMENT\_METHOD |  | |  | CHAR | 10 | |
| 10 | DUE\_DATE |  | |  | CHAR | 10 | |
| 11 | PAYMENT\_STATUS |  | |  | CHAR | 1 | |
| 12 | UPDATE\_TIMESTAMP |  | |  | CHAR | 26 | |
| 13 | INSERT\_TIMESTAMP |  | |  | CHAR | 26 | |
| 14 | PRODUCT\_CODE |  | |  | CHAR | 8 | |

### [CRDT.PREPAID\_REQ]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **CRDT.PREPAID\_REQ** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | BUSINESS\_DATE |  | |  | CHAR | 10 | |
| 2 | SEQ\_NO |  | |  | CHAR | 3 | |
| 3 | TYPE |  | |  | CHAR | 2 | |
| 4 | ORG\_CDE |  | |  | CHAR | 3 | |
| 5 | LE\_SYDIPEL |  | |  | DEC FIXED | (11,0) | |
| 6 | LE\_DEP\_ACC |  | |  | CHAR | 16 | |
| 7 | HOLD\_KEY\_AMOUNT |  | |  | CHAR | 26 | |
| 8 | HOLD\_AMOUNT |  | |  | DEC FIXED | (15,2) | |
| 9 | DEB\_AMOUNT |  | |  | CHAR | 1 | |
| 10 | HOLD\_KEY\_FEE |  | |  | CHAR | 26 | |
| 11 | HOLD\_FEE |  | |  | DEC FIXED | (15,2) | |
| 12 | DEB\_FEE |  | |  | CHAR | 1 | |
| 13 | FINAL\_AMOUNT |  | |  | DEC FIXED | (15,2) | |
| 14 | FINAL\_FEE |  | |  | DEC FIXED | (15,2) | |
| 15 | TOTAL\_REC\_COUNT |  | |  | DEC FIXED | (8,0) | |
| 16 | SUCC\_REC\_COUNT |  | |  | DEC FIXED | (8,0) | |
| 17 | REQUEST\_STATUS |  | |  | CHAR | 1 | |
| 18 | RC |  | |  | CHAR | 2 | |
| 19 | REJECTION\_TXT |  | |  | CHAR | 80 | |
| 20 | INSERT\_TMSTMP |  | |  | CHAR | 26 | |

### [CRDT.PREPAID\_REQ\_LOA]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **CRDT.PREPAID\_REQ\_LOA** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | BUSINESS\_DATE |  | |  | CHAR | 10 | |
| 2 | SEQ\_NO |  | |  | CHAR | 3 | |
| 3 | TYPE |  | |  | CHAR | 2 | |
| 4 | ORG\_CDE |  | |  | CHAR | 3 | |
| 5 | CARD\_ID |  | |  | CHAR | 20 | |
| 6 | LE\_SYDIPEL |  | |  | DEC FIXED | (11,0) | |
| 7 | LE\_DEP\_ACC |  | |  | CHAR | 16 | |
| 8 | REQUEST\_STATUS |  | |  | CHAR | 1 | |
| 9 | AMOUNT |  | |  | DEC FIXED | (15,2) | |
| 10 | FEE |  | |  | DEC FIXED | (15,2) | |
| 11 | COMM\_LOG\_ID |  | |  | CHAR | 26 | |
| 12 | IBAN |  | |  | CHAR | 27 | |
| 13 | RC |  | |  | CHAR | 2 | |
| 14 | REJECTION\_TXT |  | |  | CHAR | 80 | |
| 15 | INSERT\_TMSTMP |  | |  | CHAR | 26 | |

### [CRDT.PREPAID\_REQ\_LOA\_HIST]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **CRDT.PREPAID\_REQ\_LOA\_HIST** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | BUSINESS\_DATE |  | |  | CHAR | 10 | |
| 2 | SEQ\_NO |  | |  | CHAR | 3 | |
| 3 | TYPE |  | |  | CHAR | 2 | |
| 4 | ORG\_CDE |  | |  | CHAR | 3 | |
| 5 | CARD\_ID |  | |  | CHAR | 20 | |
| 6 | LE\_SYDIPEL |  | |  | DEC FIXED | (11,0) | |
| 7 | LE\_DEP\_ACC |  | |  | CHAR | 16 | |
| 8 | REQUEST\_STATUS |  | |  | CHAR | 1 | |
| 9 | AMOUNT |  | |  | DEC FIXED | (15,2) | |
| 10 | FEE |  | |  | DEC FIXED | (15,2) | |
| 11 | COMM\_LOG\_ID |  | |  | CHAR | 26 | |
| 12 | IBAN |  | |  | CHAR | 27 | |
| 13 | RC |  | |  | CHAR | 2 | |
| 14 | REJECTION\_TXT |  | |  | CHAR | 80 | |
| 15 | INSERT\_TMSTMP |  | |  | CHAR | 26 | |

### [CRDT.PREPAID\_REQ\_ISS]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **CRDT.PREPAID\_REQ\_ISS** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | BUSINESS\_DATE |  | |  | CHAR | 10 | |
| 2 | SEQ\_NO |  | |  | CHAR | 3 | |
| 3 | TYPE |  | |  | CHAR | 2 | |
| 4 | ORG\_CDE |  | |  | CHAR | 3 | |
| 5 | UNIQUE\_ID |  | |  | CHAR | 27 | |
| 6 | PRODUCTION\_EVENT |  | |  | CHAR | 3 | |
| 7 | AMOUNT |  | |  | DEC FIXED | (15,2) | |
| 8 | FEE |  | |  | DEC FIXED | (15,2) | |
| 9 | COMM\_LOG\_ID |  | |  | CHAR | 26 | |
| 10 | PAN |  | |  | CHAR | 19 | |
| 11 | PROD\_CODE |  | |  | CHAR | 3 | |
| 12 | NEW\_PAN |  | |  | CHAR | 19 | |
| 13 | NEW\_PRD\_CODE |  | |  | CHAR | 3 | |
| 14 | CRD\_EXP\_DAT |  | |  | CHAR | 10 | |
| 15 | PROD\_NAME |  | |  | CHAR | 30 | |
| 16 | PROD\_GROUP |  | |  | CHAR | 1 | |
| 17 | SMARTPIN\_IND |  | |  | CHAR | 1 | |
| 18 | LAST\_NAME |  | |  | CHAR | 40 | |
| 19 | FIRST\_NAME |  | |  | CHAR | 40 | |
| 20 | FATHER\_NAME |  | |  | CHAR | 40 | |
| 21 | BIRTH\_DATE |  | |  | CHAR | 10 | |
| 22 | ID\_TYPE |  | |  | CHAR | 2 | |
| 23 | ID\_NO |  | |  | CHAR | 15 | |
| 24 | ID\_ISS\_DATE |  | |  | CHAR | 10 | |
| 25 | ID\_EXP\_DATE |  | |  | CHAR | 10 | |
| 26 | ID\_ISS\_CTRY |  | |  | CHAR | 2 | |
| 27 | CITIZENSHIP |  | |  | CHAR | 2 | |
| 28 | TAX\_ID\_NO |  | |  | CHAR | 9 | |
| 29 | EMPLOYEE\_ID |  | |  | CHAR | 16 | |
| 30 | PE\_IBAN |  | |  | CHAR | 27 | |
| 31 | CARD\_ID |  | |  | CHAR | 20 | |
| 32 | DEPARTMENT\_ID |  | |  | CHAR | 10 | |
| 33 | LE\_IBAN |  | |  | CHAR | 27 | |
| 34 | LE\_DEP\_ACC |  | |  | CHAR | 16 | |
| 35 | LE\_SIDIPEL |  | |  | DEC FIXED | (11,0) | |
| 36 | PE\_SIDIPEL |  | |  | DEC FIXED | (11,0) | |
| 37 | COMPANY\_NAME |  | |  | CHAR | 27 | |
| 38 | COMPANY\_ADDR\_1 |  | |  | CHAR | 40 | |
| 39 | COMPANY\_ADDR\_2 |  | |  | CHAR | 40 | |
| 40 | COMPANY\_ADDR\_3 |  | |  | CHAR | 40 | |
| 41 | COMPANY\_ADDR\_4 |  | |  | CHAR | 40 | |
| 42 | REQUEST\_STATUS |  | |  | CHAR | 1 | |
| 43 | RC |  | |  | CHAR | 2 | |
| 44 | RC |  | |  | CHAR | 80 | |
| 45 | INSERT\_TMSTMP |  | |  | CHAR | 26 | |
| 46 | MOBILE |  | |  | CHAR | 10 | |
| 47 | PE\_SIDIPEL\_FLAG |  | |  | CHAR | 2 | |

### [CRDT.PREPAID\_REQ\_ISS\_HIST]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **CRDT.PREPAID\_REQ\_ISS\_HIST** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | BUSINESS\_DATE |  | |  | CHAR | 10 | |
| 2 | SEQ\_NO |  | |  | CHAR | 3 | |
| 3 | TYPE |  | |  | CHAR | 2 | |
| 4 | ORG\_CDE |  | |  | CHAR | 3 | |
| 5 | UNIQUE\_ID |  | |  | CHAR | 27 | |
| 6 | PRODUCTION\_EVENT |  | |  | CHAR | 3 | |
| 7 | AMOUNT |  | |  | DEC FIXED | (15,2) | |
| 8 | FEE |  | |  | DEC FIXED | (15,2) | |
| 9 | COMM\_LOG\_ID |  | |  | CHAR | 26 | |
| 10 | PAN |  | |  | CHAR | 19 | |
| 11 | PROD\_CODE |  | |  | CHAR | 3 | |
| 12 | NEW\_PAN |  | |  | CHAR | 19 | |
| 13 | NEW\_PRD\_CODE |  | |  | CHAR | 3 | |
| 14 | CRD\_EXP\_DAT |  | |  | CHAR | 10 | |
| 15 | PROD\_NAME |  | |  | CHAR | 30 | |
| 16 | PROD\_GROUP |  | |  | CHAR | 1 | |
| 17 | SMARTPIN\_IND |  | |  | CHAR | 1 | |
| 18 | LAST\_NAME |  | |  | CHAR | 40 | |
| 19 | FIRST\_NAME |  | |  | CHAR | 40 | |
| 20 | FATHER\_NAME |  | |  | CHAR | 40 | |
| 21 | BIRTH\_DATE |  | |  | CHAR | 10 | |
| 22 | ID\_TYPE |  | |  | CHAR | 2 | |
| 23 | ID\_NO |  | |  | CHAR | 15 | |
| 24 | ID\_ISS\_DATE |  | |  | CHAR | 10 | |
| 25 | ID\_EXP\_DATE |  | |  | CHAR | 10 | |
| 26 | ID\_ISS\_CTRY |  | |  | CHAR | 2 | |
| 27 | CITIZENSHIP |  | |  | CHAR | 2 | |
| 28 | TAX\_ID\_NO |  | |  | CHAR | 9 | |
| 29 | EMPLOYEE\_ID |  | |  | CHAR | 16 | |
| 30 | PE\_IBAN |  | |  | CHAR | 27 | |
| 31 | CARD\_ID |  | |  | CHAR | 20 | |
| 32 | DEPARTMENT\_ID |  | |  | CHAR | 10 | |
| 33 | LE\_IBAN |  | |  | CHAR | 27 | |
| 34 | LE\_DEP\_ACC |  | |  | CHAR | 16 | |
| 35 | LE\_SIDIPEL |  | |  | DEC FIXED | (11,0) | |
| 36 | PE\_SIDIPEL |  | |  | DEC FIXED | (11,0) | |
| 37 | COMPANY\_NAME |  | |  | CHAR | 27 | |
| 38 | COMPANY\_ADDR\_1 |  | |  | CHAR | 40 | |
| 39 | COMPANY\_ADDR\_2 |  | |  | CHAR | 40 | |
| 40 | COMPANY\_ADDR\_3 |  | |  | CHAR | 40 | |
| 41 | COMPANY\_ADDR\_4 |  | |  | CHAR | 40 | |
| 42 | REQUEST\_STATUS |  | |  | CHAR | 1 | |
| 43 | RC |  | |  | CHAR | 2 | |
| 44 | RC |  | |  | CHAR | 80 | |
| 45 | INSERT\_TMSTMP |  | |  | CHAR | 26 | |
| 46 | MOBILE |  | |  | CHAR | 10 | |
| 47 | PE\_SIDIPEL\_FLAG |  | |  | CHAR | 2 | |

### [KCCT.COMMISSIONS\_TRN]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **KCCT.COMMISIONS\_TRN** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | ID |  | |  | CHAR | 18 | |
| 2 | CARD |  | |  | CHAR | 16 | |
| 3 | SUPPLY\_IND |  | |  | CHAR | 8 | |
| 4 | FREE\_IND |  | |  | CHAR | 1 | |
| 5 | FREE\_CODE |  | |  | CHAR | 2 | |
| 6 | AMOUNT |  | |  | DEC FIXED | (15,2) | |
| 7 | PMT\_METHOD |  | |  | CHAR | 4 | |
| 8 | ACCOUNT |  | |  | CHAR | 16 | |
| 9 | DESM\_ID |  | |  | CHAR | 20 | |
| 10 | CREATED |  | |  | CHAR | 26 | |
| 11 | UPDATED |  | |  | CHAR | 26 | |
| 12 | STATUS |  | |  | CHAR | 1 | |
| 13 | BRANCH |  | |  | CHAR | 5 | |

### [KCCT.SMARTPIN]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **KCCT.SMARTPIN** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | CARD |  | |  | CHAR | 16 | |
| 2 | PRODUCT |  | |  | CHAR | 9 | |
| 3 | CARD\_ORIGIN |  | |  | CHAR | 2 | |
| 4 | CHANNEL |  | |  | CHAR | 3 | |
| 5 | REQUEST |  | |  | CHAR | 1 | |
| 6 | FLG\_DELIVER |  | |  | CHAR | 1 | |
| 7 | CRA\_STRC\_K |  | |  | DEC FIXED | (11,0) | |
| 8 | CRA\_ADDR |  | |  | CHAR | 40 | |
| 9 | CRA\_TK |  | |  | CHAR | 40 | |
| 10 | CRA\_CITY |  | |  | CHAR | 40 | |
| 11 | CRA\_COUNTRY |  | |  | CHAR | 40 | |
| 12 | CRA\_EMAIL |  | |  | CHAR | 40 | |
| 13 | CRA\_PHONE |  | |  | CHAR | 40 | |
| 14 | MATRIX |  | |  | CHAR | 20 | |
| 15 | REQ\_TMSTMP |  | |  | CHAR | 26 | |
| 16 | INS\_TMSTMP |  | |  | CHAR | 26 | |
| 17 | UPD\_TMSTMP |  | |  | CHAR | 26 | |
| 18 | FLG |  | |  | CHAR | 1 | |

### [KCCT.SMARTPIN\_HIST]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **KCCT.SMARTPIN\_HIST** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | CARD |  | |  | CHAR | 16 | |
| 2 | PRODUCT |  | |  | CHAR | 9 | |
| 3 | CARD\_ORIGIN |  | |  | CHAR | 2 | |
| 4 | CHANNEL |  | |  | CHAR | 3 | |
| 5 | REQUEST |  | |  | CHAR | 1 | |
| 6 | FLG\_DELIVER |  | |  | CHAR | 1 | |
| 7 | CRA\_STRC\_K |  | |  | DEC FIXED | (11,0) | |
| 8 | CRA\_ADDR |  | |  | CHAR | 40 | |
| 9 | CRA\_TK |  | |  | CHAR | 40 | |
| 10 | CRA\_CITY |  | |  | CHAR | 40 | |
| 11 | CRA\_COUNTRY |  | |  | CHAR | 40 | |
| 12 | CRA\_EMAIL |  | |  | CHAR | 40 | |
| 13 | CRA\_PHONE |  | |  | CHAR | 40 | |
| 14 | MATRIX |  | |  | CHAR | 20 | |
| 15 | REQ\_TMSTMP |  | |  | CHAR | 26 | |
| 16 | INS\_TMSTMP |  | |  | CHAR | 26 | |
| 17 | UPD\_TMSTMP |  | |  | CHAR | 26 | |
| 18 | FLG |  | |  | CHAR | 1 | |

### [CRDT.MSG\_UNIQ\_ARC]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **CRDT.MSG\_UNIQ\_ARC** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | SRN |  | |  | CHAR | 12 | |
| 2 | TRDATE |  | |  | CHAR | 10 | |
| 3 | MSG\_TYPE |  | |  | CHAR | 4 | |
| 4 | TRN\_CDE |  | |  | CHAR | 3 | |

### [CRDT.MSG\_UNIQ\_SESP\_HIST\_ARC]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **CRDT.MSG\_UNIQ\_SESP\_HIST\_ARC** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | OR\_TYP |  | |  | CHAR | 4 | |
| 2 | OR\_STAN |  | |  | CHAR | 6 | |
| 3 | OR\_TRDATE |  | |  | CHAR | 10 | |
| 4 | OR\_ACQID |  | |  | CHAR | 6 | |
| 5 | RRN |  | |  | CHAR | 12 | |
| 6 | ACQID |  | |  | CHAR | 6 | |
| 7 | CARD |  | |  | CHAR | 10 | |
| 8 | TYP |  | |  | CHAR | 4 | |
| 9 | STAN |  | |  | CHAR | 6 | |
| 10 | AUTHCODE |  | |  | CHAR | 6 | |
| 11 | UPDATE\_TIMESTAMP |  | |  | CHAR | 26 | |

### [KCCT.PR\_ADDRS]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **KCCT.PR\_ADDRS** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | ACCOUNT |  | |  | CHAR | 13 | |
| 2 | PLUS\_ADDR |  | |  | CHAR | 30 | |
| 3 | HOME\_ADDR |  | |  | CHAR | 24 | |
| 4 | HOME\_ZIPCODE |  | |  | CHAR | 5 | |
| 5 | HOME\_CITY |  | |  | CHAR | 16 | |
| 6 | HOME\_CODETEL |  | |  | CHAR | 5 | |
| 7 | HOME\_TEL |  | |  | CHAR | 7 | |
| 8 | JOB\_ADDR |  | |  | CHAR | 24 | |
| 9 | JOB\_ZIPCODE |  | |  | CHAR | 5 | |
| 10 | JOB\_CITY |  | |  | CHAR | 16 | |
| 11 | JOB\_CODETEL |  | |  | CHAR | 5 | |
| 12 | JOB\_TEL |  | |  | CHAR | 7 | |
| 13 | MOBILE |  | |  | CHAR | 10 | |
| 14 | WHERE\_FLG |  | |  | CHAR | 2 | |
| 15 | UPD\_IND |  | |  | CHAR | 1 | |
| 16 | STMT\_EMAIL |  | |  | CHAR | 80 | |

### [CRDT.DEB\_FILE\_EXC]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **CRDT.DEB\_FILE\_EXC** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | PGM |  | |  | CHAR | 30 | |
| 2 | MULT\_EXEC |  | |  | CHAR | 1 | |
| 3 | LATST\_TMSTMP |  | |  | CHAR | 26 | |

### [CRDT.MASS\_ISS\_REQ]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **CRDT.MASS\_ISS\_REQ** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | APPL\_NUM |  | |  | CHAR | 20 | |
| 2 | APPL\_STATUS |  | |  | CHAR | 1 | |
| 3 | APPL\_REQ |  | |  | CHAR | 3 | |
| 4 | SYDIPEL |  | |  | DEC FIXED | (11,0) | |
| 5 | CLS\_CLIENT |  | |  | CHAR | 8 | |
| 6 | DEP\_ACC |  | |  | CHAR | 16 | |
| 7 | ACC\_TYPE |  | |  | CHAR | 2 | |
| 8 | CRD\_TYPE |  | |  | CHAR | 2 | |
| 9 | PROD\_CODE |  | |  | CHAR | 9 | |
| 10 | PROD\_GROUP |  | |  | CHAR | 2 | |
| 11 | FIRST\_NAME |  | |  | CHAR | 32 | |
| 12 | LAST\_NAME |  | |  | CHAR | 32 | |
| 13 | COMP\_NAME |  | |  | CHAR | 32 | |
| 14 | ISSUE\_BRANCH |  | |  | CHAR | 3 | |
| 15 | PIN\_BRANCH |  | |  | CHAR | 3 | |
| 16 | CARDS\_REC |  | |  | DEC FIXED | (5,0) | |
| 17 | TRAN\_DATE |  | |  | CHAR | 10 | |
| 18 | TRAN\_TIME |  | |  | CHAR | 8 | |
| 19 | BATCH\_DATE |  | |  | CHAR | 10 | |
| 20 | APPL\_TIMESTAMP |  | |  | CHAR | 26 | |
| 21 | APPL\_TIMESTAMP\_UPD |  | |  | CHAR | 26 | |
| 22 | QUANTITY\_COMM |  | |  | CHAR | 30 | |
| 23 | QUALITY\_COMM |  | |  | CHAR | 30 | |

### [KCCT.TRACK\_INFO]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **KCCT.TRACK\_INFO** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | ID |  | |  | CHAR | 20 | |
| 2 | RECORD\_NUMBER |  | |  | CHAR | 6 | |
| 3 | HOLDER |  | |  | CHAR | 26 | |
| 4 | CARD |  | |  | CHAR | 16 | |
| 5 | REJECTION\_CODE |  | |  | CHAR | 3 | |
| 6 | REJECTION\_DESC |  | |  | CHAR | 100 | |
| 7 | FILE\_NAME |  | |  | CHAR | 15 | |
| 8 | STATUS |  | |  | CHAR | 3 | |
| 9 | STATUS\_DATE |  | |  | CHAR | 10 | |
| 10 | STREET |  | |  | CHAR | 40 | |
| 11 | CITY |  | |  | CHAR | 40 | |
| 12 | ZIPCODE |  | |  | CHAR | 20 | |
| 13 | COUNTRY |  | |  | CHAR | 40 | |
| 14 | PHONE |  | |  | CHAR | 14 | |
| 15 | DESTINATION |  | |  | CHAR | 1 | |
| 16 | SHIP\_METHOD |  | |  | CHAR | 1 | |
| 17 | BARCODE |  | |  | CHAR | 16 | |
| 18 | USERID |  | |  | CHAR | 10 | |
| 19 | CHANNEL |  | |  | CHAR | 3 | |
| 20 | TMSTMP |  | |  | CHAR | 26 | |
| 21 | EMBOSSING\_DATE |  | |  | CHAR | 10 | |
| 22 | SENT\_DATE |  | |  | CHAR | 10 | |
| 23 | KIND |  | |  | CHAR | 3 | |
| 24 | PROD\_STATUS |  | |  | CHAR | 1 | |
| 25 | PROD\_STATUS\_DATE |  | |  | CHAR | 10 | |
| 26 | PRODUCT\_CODE |  | |  | CHAR | 3 | |
| 27 | PRODUCT\_NAME |  | |  | CHAR | 30 | |
| 28 | TRACKING\_NUM |  | |  | CHAR | 20 | |

### [CRDT.LOADING\_MASTER]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **CRDT.LOADING\_MASTER** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | BUSINESS\_DATE |  | |  | CHAR | 10 | |
| 2 | SEQUENCE\_NUMBER |  | |  | CHAR | 10 | |
| 3 | ORGANIZATION\_CODE |  | |  | CHAR | 3 | |
| 4 | TYPE |  | |  | CHAR | 2 | |
| 5 | LEGAL\_SYDIPEL |  | |  | DEC FIXED | (11,0) | |
| 6 | INSTITUTION\_D\_ACCOUNT |  | |  | CHAR | 26 | |
| 7 | HOLD\_KEY\_AMOUNT |  | |  | CHAR | 26 | |
| 8 | HOLD\_KEY\_FEE |  | |  | CHAR | 26 | |
| 9 | HOLD\_AMOUNT |  | |  | DEC FIXED | (15,2) | |
| 10 | HOLD\_FEE |  | |  | DEC FIXED | (15,2) | |
| 11 | FINAL\_AMOUNT |  | |  | DEC FIXED | (15,2) | |
| 12 | FINAL\_FEE |  | |  | DEC FIXED | (15,2) | |
| 13 | DEB\_AMOUNT |  | |  | CHAR | 1 | |
| 14 | DEB\_FEE |  | |  | CHAR | 1 | |
| 15 | STATUS |  | |  | CHAR | 2 | |
| 16 | VALID\_REC\_COUNT |  | |  | DEC FIXED | (8,0) | |
| 17 | SUC\_REC\_COUNT |  | |  | DEC FIXED | (8,0) | |
| 18 | RC\_TXT |  | |  | CHAR | 80 | |
| 19 | INSERT\_TMSTMP |  | |  | CHAR | 26 | |

### [KCCT.FILE\_TRAN]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **KCCT.FILE\_TRAN** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | ID |  | |  | CHAR | 20 | |
| 2 | APPL |  | |  | CHAR | 8 | |
| 3 | ROUTE |  | |  | CHAR | 8 | |
| 4 | SUB\_APPL |  | |  | CHAR | 8 | |
| 5 | NAME |  | |  | CHAR | 40 | |
| 6 | CREATED |  | |  | CHAR | 26 | |
| 7 | NBOFTXS |  | |  | BIN FIXED | 31 | |

### [CRDT.EMBOS\_PARM]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **CRDT.EMBOS\_PARM** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | APPLICATION |  | |  | CHAR | 8 | |
| 2 | PRODUCTION\_CODE |  | |  | CHAR | 1 | |
| 3 | PRODUCT\_CODE |  | |  | CHAR | 3 | |
| 4 | SUFFIX |  | |  | CHAR | 8 | |
| 5 | TEMPLATE\_NAME |  | |  | CHAR | 12 | |
| 6 | LYKOS\_PREFIX |  | |  | CHAR | 3 | |
| 7 | RWL\_MIG\_FLAG |  | |  | CHAR | 1 | |

### [CRDT.EMBOS\_COUNTER]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **CRDT.EMBOS\_COUNTER** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | APPLICATION |  | |  | CHAR | 8 | |
| 2 | SUB\_APPLICATION |  | |  | CHAR | 8 | |
| 3 | LAST\_CNT\_USED |  | |  | DEC FIXED | (5,0) | |

### [CRDT.EMBOS\_FILE\_MASTER]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **CRDT.EMBOS\_FILE\_MASTER** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | FILE\_ID |  | |  | DEC FIXED | (10,0) | |
| 2 | FILE\_DATE |  | |  | CHAR | 10 | |
| 3 | FILE\_TIME |  | |  | CHAR | 8 | |
| 4 | FILE\_NUMBER |  | |  | CHAR | 6 | |
| 5 | PRODUCTION\_CODE |  | |  | CHAR | 1 | |
| 6 | PRODUCT\_CODE |  | |  | CHAR | 3 | |
| 7 | WAY4\_FILE\_NAME |  | |  | CHAR | 40 | |
| 8 | LYKOS\_FILE\_NAME |  | |  | CHAR | 15 | |
| 9 | FILE\_STATUS |  | |  | CHAR | 1 | |
| 10 | INS\_TMSTMP |  | |  | CHAR | 26 | |
| 11 | UPD\_TMSTMP |  | |  | CHAR | 26 | |
| 12 | TRANSACTION\_ID |  | |  | CHAR | 255 | |

### [CRDT.EMBOS\_FILE\_DETAIL]

| **Table Information** | | | | | | |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Table name** | | | **CRDT.EMBOS\_FILE\_DETAIL** | | | |  |
| **Table description** | | |  | | | |  |
| **Database** | | | Mainframe | | | |  |
|  | | |  | | | |  |
| **Input Data** | | | | | | | |
| **A/A** | **Name** | **Description** | | **Primary Key** | **Data Type** | **Length** | |
| 1 | FILE\_ID |  | |  | DEC FIXED | (10,0) | |
| 2 | BATCH\_ID |  | |  | CHAR | 32 | |
| 3 | REC\_STATUS |  | |  | CHAR | 2 | |
| 4 | CARD |  | |  | CHAR | 19 | |
| 5 | PRODUCTION\_EVENT |  | |  | CHAR | 4 | |
| 6 | SVC |  | |  | CHAR | 3 | |
| 7 | PL\_DELIVERY\_TYPE |  | |  | CHAR | 2 | |
| 8 | PIN\_DELIVERY\_BRANCH |  | |  | CHAR | 5 | |
| 9 | ISSUING\_BRANCH |  | |  | CHAR | 5 | |
| 10 | PE\_EMBOSSING\_NAME |  | |  | CHAR | 26 | |
| 11 | EMBOSSING\_NAME |  | |  | CHAR | 26 | |
| 12 | EXPIRATION\_DATE |  | |  | CHAR | 10 | |
| 13 | TOTAL\_CASH\_WTH\_LMT |  | |  | DEC FIXED | (15,2) | |
| 14 | POS\_PURCHASE\_LMT |  | |  | DEC FIXED | (15,2) | |
| 15 | TOTAL\_CNP\_LMT |  | |  | DEC FIXED | (15,2) | |
| 16 | ISUUING\_DATE |  | |  | CHAR | 10 | |
| 17 | CARD\_STATUS |  | |  | CHAR | 1 | |
| 18 | PIN\_PROD\_STATUS |  | |  | CHAR | 1 | |
| 19 | PLA\_PROD\_STATUS |  | |  | CHAR | 1 | |
| 20 | PLA\_STATUS |  | |  | CHAR | 1 | |
| 21 | PRODUCT\_CODE |  | |  | CHAR | 3 | |
| 22 | PRODUCT\_GROUP |  | |  | CHAR | 2 | |
| 23 | INSTITUTION\_ID |  | |  | CHAR | 4 | |
| 24 | SMARTPIN |  | |  | CHAR | 1 | |
| 25 | PRODUCT\_NAME |  | |  | CHAR | 30 | |
| 26 | RESIDENT |  | |  | CHAR | 1 | |
| 27 | PERSONAL\_TYPE |  | |  | CHAR | 2 | |
| 28 | CLASS\_PRODUCT |  | |  | CHAR | 2 | |
| 29 | KIND |  | |  | CHAR | 3 | |
| 30 | COLLECTION\_DATE |  | |  | CHAR | 10 | |
| 31 | MATURITY\_DATE |  | |  | CHAR | 10 | |
| 32 | CONDITION |  | |  | DEC FIXED | (2,0) | |
| 33 | SNAME |  | |  | CHAR | 40 | |
| 34 | FNAME |  | |  | CHAR | 40 | |
| 35 | PNAME |  | |  | CHAR | 40 | |
| 36 | COUNTRY |  | |  | CHAR | 40 | |
| 37 | STREET |  | |  | CHAR | 40 | |
| 38 | ZIPCODE |  | |  | CHAR | 40 | |
| 39 | CITY |  | |  | CHAR | 40 | |
| 40 | I\_IP |  | |  | DEC FIXED | (11,0) | |
| 41 | PVV |  | |  | CHAR | 4 | |
| 42 | CVV1 |  | |  | DEC FIXED | (3,0) | |
| 43 | CVV2 |  | |  | DEC FIXED | (3,0) | |
| 44 | ICVV |  | |  | DEC FIXED | (3,0) | |
| 45 | I\_IP\_ET\_KAT |  | |  | DEC FIXED | (11,0) | |
| 46 | PIN\_BLOCK |  | |  | CHAR | 16 | |
| 47 | REJECT\_REASON |  | |  | CHAR | 2 | |
| 48 | REJECT\_REASON\_TXT |  | |  | CHAR | 50 | |
| 49 | UPD\_TMSTMP |  | |  | CHAR | 26 | |
| 50 | CARD\_ENR |  | |  | CHAR | 19 | |
| 51 | DOM\_CASH\_WTH\_LMT |  | |  | DEC FIXED | (15,2) | |
| 52 | INT\_CASH\_WTH\_LMT |  | |  | DEC FIXED | (15,2) | |
| 53 | DOM\_CNP\_LMT |  | |  | DEC FIXED | (15,2) | |
| 54 | INT\_CNP\_LMT |  | |  | DEC FIXED | (15,2) | |
| 55 | SCHEME |  | |  | CHAR | 10 | |
| 56 | BARCODE |  | |  | CHAR | 20 | |
| 57 | EMPLOYEE\_ID |  | |  | CHAR | 20 | |
| 58 | DEPARTMENT\_ID |  | |  | CHAR | 20 | |
| 59 | IBAN |  | |  | CHAR | 27 | |
| 60 | CARD\_TYPE |  | |  | CHAR | 4 | |
| 61 | COLOR |  | |  | CHAR | 4 | |
| 62 | COMBO\_CARD |  | |  | CHAR | 19 | |

### 