RV001R.MBR

Path: NXCLOUD/rpgsrc/RV001R.MBR **Generated**: 2026-01-08 12:59:00 **Processing Time**: 13118ms

Business Logic for RV001R

This document outlines the business rules that govern the processing of financial records, specifically reading lines from the FOVF file and converting them into a comma-separated format, based on an analysis of the RPG program RV001R. The primary focus is on the data transformation and validation logic.

The core logic for processing financial records is contained within the main program RV001R. The program reads data from various input files, performs necessary transformations, and writes the output to a defined output file.

Order Status and Header Rules

RV001R: FOVFL1, SOHEL1, VOTYL1, AFORL1, RB10L1, RkunL1, RB00L1

1. Firm Check

- **Logic:** The program checks if the current firm (l_firm) matches the firm in the record (fffirm). If they do not match, the program skips processing for that record.
- **File:** FOVFL1 (Input file for financial records)
- **Field:** l_firm
- **Condition:** "The process will not select a record if l_firm is not equal to fffirm."

2. MVA Code Assignment

- **Logic:** The program assigns a MVA code based on whether the field ffdmom is blank. If it is blank, it uses ffcmom; otherwise, it uses ffdmom.
- **File:** FOVFL1
- **Field:** ffdmom
- **Condition:** "If ffdmom is blank, assign ffcmom to w_mkod; otherwise, assign ffdmom."

Configuration and Authorization Rules

1. Invoice Type Conversion

- **Logic:** The program checks if the invoice type (ffbilk) needs conversion by chaining to the RB10L1 file. If found, it updates the invoice type.
- **Files:**
- RB10L1 (Invoice type mapping)
- **Fields:**
- ffbilk (Original invoice type)
- rb10l1_gkod (Converted invoice type)
- **Condition:** "If the invoice type exists in RB10L1, it will be converted."

2. Customer Data Retrieval

- **Logic:** The program retrieves customer data from the RkunL1 file based on the customer number (solkun).
- **File:** RkunL1 (Customer data)

•**Field:** solkun
•**Condition:** "Customer data is fetched if solkun is not zero."

Financial and Transactional Rules

1. Posting Date Assignment
•**Logic:** The program assigns the posting date from the record to the output structure.
•**File:** FOVFL1
•**Fields:**
•ffbdat (Posting date from input)
•w_dato_al1 (Posting date in output)
•**Condition:** "The posting date is directly assigned from ffbdat."
2. Record Writing
•**Logic:** The program constructs a comma-separated string of various fields and writes it to the output file RV01PFR.
•**File:** RV01PFR (Output file)
•**Condition:** "The record is written after all fields are processed and formatted."

Special Conditions (Program-Specific)

1. Comma Check in Names (RV001R)
•**Logic:** The program checks if the delivery name or customer name contains a comma. If so, it splits the name into two parts.
•**File:** RkunL1
•**Field:** rknavn
•**Condition:** "If rknavn contains a comma, it is split into first and last names."
2. Handling of Blank Fields (RV001R)
•**Logic:** The program initializes several fields to blank if the firm check fails.
•**File:** FOVFL1
•**Fields:** Various fields like w_lnavn, w_ladr1, etc.
•**Condition:** "Fields are set to blank if the firm does not match."

Subprogram Calls Affecting Logic

Beyond direct file checks, several external subprograms are called that play a significant role in the workflow.

1. control_init (Initialization Subroutine)
•**Trigger:** Called at the beginning of the record processing.
•**Logic:** Initializes key variables and retrieves the latest invoice header for processing.
•**Impact:** "This call sets up the necessary context for processing each record."
2. AK710R (Routine for Handling Customer Data)
•**Trigger:** Called to retrieve additional customer information based on specific conditions.
•**Logic:** Fetches customer-related data based on the provided keys.
•**Impact:** "This call enriches the data being processed with customer-specific details."
3. %scan (String Function)

•**Trigger:** Used multiple times to check for commas in names.
•**Logic:** Searches for the position of a comma in a string.
•**Impact:** "This function is crucial for determining how to split names for output formatting."