RS001R.MBR

Path: NXCLOUD/rpgsrc/RS001R.MBR **Generated**: 2026-01-08 12:35:21 **Processing Time**: 9794ms

Business Logic for Customer Number Retrieval

This document outlines the business rules that govern the retrieval of the last used customer number, based on an analysis of the RPG program RS001R. The primary focus is on how the program determines and updates the last used customer number based on various conditions. The core logic for retrieving the last used customer number is contained within the main processing logic of the program. The program checks for the existence of the last used customer number and updates it based on specific business rules.

Customer Number and Validation Rules

Customer Number Retrieval: raa4pfr, rkunpfr, raa1pfr

1. Check Last Used Customer Number

•**Logic:** The program attempts to retrieve the last used customer number from the raa4l1 file. If not found, it sets a return flag to indicate failure.

•**File:** raa4l1 (Last Used Customer Number File)

•**Field:** ra4knr

•**Condition:** The process will not select a record if the last used customer number is not found (*in90 is set to *on).

2. Update Last Used Customer Number

•**Logic:** The program updates the last used customer number based on the provided parameters and checks against various conditions.

•**File:** raa4l1 (Last Used Customer Number File)

•**Field:** ra4knr

•**Condition:** If the w_test variable is blank, the last used customer number is incremented; if it equals 'S', it is decremented; if it equals 'K', it is set to a specific parameter.

Boundary and Existence Checks

1. Boundary Check for Customer Number

•**Logic:** The program checks if the updated customer number is within valid boundaries defined in the raa1l1 file.

•**Files:**

•raa1l1 (Boundary Values File)

•**Fields:**

•ra1hkt (Lower Boundary)

•ra1hrs (Upper Boundary)

•**Condition:** The process will set the return flag to 'N' if the updated customer number is less than or equal to the lower boundary or greater than or equal to the upper boundary.

2. Check Existence of Customer Number

•**Logic:** The program checks if the updated customer number exists in the customer register. If it does not exist, it increments the customer number until a valid one is found.

•**File:** rkunlr (Customer Register)
•**Field:** rkkund
•**Condition:** The program continues to check for existence until a valid customer number is found (*in90 is set to *off).

Update Logic and Finalization

1. Final Update of Last Used Customer Number
•**Logic:** If a valid customer number is found, the program updates the last used customer number in the raa4lu file.
•**File:** raa4lu (Last Used Customer Number Update File)
•**Field:** ra4knr
•**Condition:** The update occurs only if the customer number found is greater than the last used customer number.
2. Return Status
•**Logic:** The program sets the return status flag to indicate success or failure of the operation.
•**File:** N/A
•**Condition:** The return flag is set to a blank value upon successful completion of the process.

Subprogram Calls Affecting Logic

Beyond direct file checks, several external subprograms are called that play a significant role in the workflow.
1. *inzsr (Initialization Subroutine)
•**Trigger:** This subroutine is called at the beginning of the program.
•**Logic:** It initializes parameters from the previous program and sets up key fields for various files.
•**Impact:** This setup is crucial for ensuring that the program has the necessary context and parameters to operate correctly.
2. *entry (Entry Point)
•**Trigger:** This is the entry point for the program, receiving parameters for processing.
•**Logic:** It takes in the return flag, firm identifier, and last used customer number as parameters.
•**Impact:** This entry point establishes the initial state and context for the program's execution.
3. chain (File Access)
•**Trigger:** The program frequently uses the chain operation to access records in various files.
•**Logic:** This operation retrieves records based on the defined keys, allowing the program to check for existence and update records.
•**Impact:** These calls are essential for the program's ability to interact with the database and retrieve necessary information.