RS001R.MBR

Path: NXCLOUD/rpgsrc/RS001R.MBR **Generated**: 2026-01-08 12:58:58 **Processing Time**: 9254ms

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 manages customer number retrieval and validation.

The core logic for retrieving the last used customer number is contained within the main program logic of RS001R. The program checks for the last used customer number, validates it against certain criteria, and updates the customer number if necessary.

Customer Number and Validation Rules

Customer Number Retrieval: raa4pfr, rkunpfr, raa1pfr

1. Initial Retrieval Check

- **Logic:** The program attempts to retrieve the last used customer number from the customer number register. If it is not found, it sets a return flag to indicate failure.
- **File:** raa4pfr (Customer Number Register)
- **Field:** ra4knr
- **Condition:** The process will not proceed if the record is not found (*in90 is set to *on).

2. Customer Number Increment Logic

- **Logic:** If the last used customer number is found, the program checks if the last used number is blank or if it needs to be incremented or decremented based on specific conditions.
- **File:** raa4pfr (Customer Number Register)
- **Field:** ra4knr
- **Condition:** The program evaluates the w_test variable to determine how to adjust the customer number.

3. Boundary Check for Customer Number

- **Logic:** The program checks if the adjusted customer number falls within predefined limits. If it does not, it sets a return flag to indicate failure.
- **Files:**
- raa1pfr (Boundary Limits Register)
- **Fields:**
- ra1hkt (Lower Limit)
- ra1hrs (Upper Limit)
- **Condition:** The process will not proceed if the adjusted customer number is less than or equal to ra1hkt or greater than or equal to ra1hrs.

4. Customer Number Availability Check

- **Logic:** The program checks if the adjusted customer number exists in the customer register. If it does, it increments the number until an available number is found.
- **File:** rkunpfr (Customer Register)

•**Field:** rkunlr_kund
•**Condition:** The program loops until a customer number is found that does not exist in the register.

5. Update Last Used Customer Number

•**Logic:** Once a valid customer number is determined, the program updates the last used customer number in the customer number register.
•**File:** raa4lu (Last Used Customer Number Register)
•**Field:** ra4knr
•**Condition:** The update occurs only if the new customer number is greater than the previously stored number.

Subprogram Calls Affecting Logic

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

1. RS001R (Customer Number Control)

•**Trigger:** This program is called to retrieve and manage the last used customer number.
•**Logic:** It processes the retrieval, validation, and updating of the customer number based on various conditions.
•**Impact:** This call acts as a **major logical gateway** for managing customer numbers, ensuring that they are unique and within specified limits.

2. *inzsr (Initialization Subroutine)

•**Trigger:** This subroutine initializes parameters and sets up keys for the program.
•**Logic:** It prepares the necessary keys for accessing the customer number and boundary limit records.
•**Impact:** This is essential for ensuring that the program can access the correct records and perform validations.

3. *entry (Entry Point Subroutine)

•**Trigger:** This subroutine is called when the program is executed.
•**Logic:** It handles the input parameters from the previous program and initializes the program state.
•**Impact:** This represents the handoff to the next major business function of retrieving and managing customer numbers.