NN700R.MBR

Path: NXCLOUD/rpgsrc/NN700R.MBR **Generated**: 2026-01-08 13:32:41 **Processing Time**: 15959ms

Business Logic for NN700R

This document outlines the business rules that govern the export of customer information from the online store, based on an analysis of the RPG program NN700R. The primary focus is on how customer data is processed and formatted for export.

The core logic for exporting customer information is contained within the main program logic of NN700R. The program retrieves customer data, organizes it into a specific format, and writes it to an output file for further processing.

Customer Data and Record Rules

NN700R: rkunl1, fkprl1, apospf, anbll3, anbhl1, netkund, netkrappr

1. Customer Existence Check

- **Logic:** The program checks if the customer exists in the rkunl1 file. If not found, it sets the status to 'E' (error) and exits.
- **File:** rkunl1 (Customer master file)
- **Field:** rkkund
- **Condition:** The process will not proceed if the customer record is not found.

2. Business vs. Private Customer Identification

- **Logic:** The program identifies if the customer is a business or a private individual. If the business number is missing, it logs a message indicating the customer will be treated as a private customer.
- **File:** rkunl1 (Customer master file)
- **Field:** rkftnr
- **Condition:** If rkftnr is equal to 0, the customer is treated as a private customer.

3. Address Validation

- **Logic:** The program checks if the address field is blank. If it is, it logs a message indicating the customer will be excluded from the export.
- **File:** rkunl1 (Customer master file)
- **Field:** rksted
- **Condition:** The process will exclude the customer if the address is blank.

Message Structure and Formatting Rules

1. Message Header Creation

- **Logic:** The program constructs the message header with a unique reference number and type ID. This header is written to the output file.
- **File:** netkund (Output file for customer data)
- **Field:** d_unhrefno, d_unhtypeid
- **Condition:** The header is always written at the beginning of the export process.

2. Customer Data Formatting

•**Logic:** The program formats customer data into a specific structure, including name, address, and contact details, and writes it to the output file.
•**File:** netkund (Output file for customer data)
•**Fields:**
•d_cusnum (Customer number)
•d_cusnam (Customer name)
•d_cusad1 (Address line 1)
•d_cusad2 (Address line 2)
•d_cuscty (Country)
•d_cusmai (Email)
•**Condition:** Customer data is written only if the customer record is valid.

Customer Project Data Rules

1. Customer Project Data Inclusion
•**Logic:** The program checks for related customer projects in the fkprl1 file and includes them in the export if they are active and valid.
•**Files:**
•fkprl1 (Customer project file)
•**Fields:**
•d_cprcun (Customer number)
•d_cprprn (Project number)
•d_cprnam (Project name)
•**Condition:** Projects are included if the project date is greater than or equal to the current date.
2. Project Address Handling
•**Logic:** If a project has a valid project number, the program retrieves the corresponding address from the apospf file.
•**File:** apospf (Postal area file)
•**Field:** apsted (Postal area)
•**Condition:** The project address is only retrieved if the project number is not zero.

Special Conditions (Program-Specific)

1. Error Handling for Missing Email
•**Logic:** If a business customer does not have an email address, the program logs a message and marks the customer for exclusion from the export.
•**File:** rkunl1 (Customer master file)
•**Field:** rkemal
•**Condition:** The process will log an exclusion message if rkemal is blank for business customers.

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:** Called at the beginning of the program to initialize keys and local data area.
•**Logic:** Sets up keys for reading customer and project data, as well as initializing the firm data.
•**Impact:** This call ensures that all necessary keys are prepared for data retrieval, which is critical for the program's execution.

2. *entry (Entry Point Subroutine)

•**Trigger:** Called when the program is executed, passing the status parameter.
•**Logic:** Initializes the program and prepares it for processing customer data.
•**Impact:** This establishes the context for the program's execution, ensuring that the status is correctly set for subsequent processing.

3. avslutt (Termination Logic)

•**Trigger:** Called at the end of the processing to clean up and exit the program.
•**Logic:** Sets the last record indicator and returns control to the calling program.
•**Impact:** This ensures that resources are properly released and the program exits gracefully.

This documentation provides a comprehensive overview of the business logic implemented in the NN700R program, detailing how customer information is processed for export.