LA101R.MBR

Path: NXCLOUD/rpgsrc/LA101R.MBR **Generated**: 2026-01-08 12:34:54 **Processing Time**: 11408ms

Business Logic for LA101R

This document outlines the business rules that govern the maintenance of status codes within the system, based on an analysis of the RPG program LA101R. The primary focus is on how the program handles the retrieval and updating of warehouse codes.

The core logic for managing warehouse codes is contained within the A1BLD subroutine in LA101R. The program processes records from the lstsl1 and lstslu files to display and update warehouse status codes based on user input.

Order Status and Header Rules

LA101R: lstsl1, lstslu

1. Retrieve Warehouse Code Information

•**Logic:** The program retrieves details from the lstsl1 file using a key and populates various fields with the corresponding values if found.

•**File:** lstsl1 (Warehouse Code Master File)

•**Field:** lstsl1_key

•**Condition:** The process will not select a record if the key does not exist in the lstsl1 file.

2. Display Warehouse Code Information

•**Logic:** After retrieving the warehouse code information, the program displays the data on the screen for user interaction.

•**File:** a1bld (Display Format for Warehouse Code)

•**Field:** Various fields populated from lstsl1

•**Condition:** The display occurs regardless of whether the record was found or not, but fields will be blank if not found.

Configuration and Authorization Rules

1. Update Warehouse Code Information

•**Logic:** The program updates the lstslu file with the values entered by the user, or writes a new record if it does not exist.

•**Files:**

•lstslu (Warehouse Code Update File)

•**Fields:**

•laabrk (Warehouse Code)

•laabin (Warehouse Name)

•**Condition:** The update occurs if the record is found; otherwise, a new record is created with the current user and timestamp.

2. User Tracking

•**Logic:** The program captures the user information and timestamps for both updates and new entries.

•**File:** lstslu (Warehouse Code Update File)
•**Field:** laaeus (User ID)
•**Condition:** User information is updated whenever a record is written or updated.

Financial and Transactional Rules

1. Timestamp Management
•**Logic:** The program records timestamps for when a warehouse code is updated or created.
•**File:** lstslu (Warehouse Code Update File)
•**Fields:**
•laaeda (Date of Update)
•laaeti (Time of Update)
•**Condition:** Timestamps are set during the update process.
2. Firm Association
•**Logic:** The program associates the warehouse code with a specific firm by setting the w_firm variable.
•**File:** lstslu (Warehouse Code Update File)
•**Condition:** The firm is set at the beginning of the program and used as part of the key for updates.

Special Conditions (Program-Specific)

1. Program Exit Logic (LA101R)
•**Logic:** The program checks for the F3 key (exit) and terminates if pressed.
•**File:** N/A
•**Field:** N/A
•**Condition:** The program will exit if the user presses the F3 key during the display.
2. Initialization Subroutine (LA101R)
•**Logic:** The initialization subroutine sets up keys for reading and updating the warehouse code.
•**File:** N/A
•**Fields:**
•w_firm (Firm Identifier)
•**Condition:** The initialization occurs at the start of the program to ensure proper key setup.

Subprogram Calls Affecting Logic

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

1. A1BLD (Display Subroutine)
•**Trigger:** Called to display the warehouse code information after retrieval.
•**Logic:** This subroutine formats and presents the warehouse code data for user interaction.
•**Impact:** This call acts as a **major logical gateway** for user input and interaction with warehouse codes.
2. *inzsr (Initialization Subroutine)
•**Trigger:** Automatically invoked at the beginning of the program.

•**Logic:** Sets up the necessary keys and initializes variables for the program's operation.
•**Impact:** This is a **critical setup step** that ensures the program has the correct context for processing.

3. end_pgm (Termination Logic)

•**Trigger:** Called when the program is ready to exit.
•**Logic:** Cleans up and sets the last record indicator before returning control.
•**Impact:** This represents the handoff to the system's main control after processing is complete.