AF010R.MBR

Path: NXCLOUD/rpgsrc/AF010R.MBR **Generated**: 2026-01-08 12:34:23 **Processing Time**: 12325ms

Business Logic for Postcode Register Maintenance

This document outlines the business rules that govern the maintenance of the postcode register, based on an analysis of the RPG program AF010R. The primary focus is on the rules and logic that dictate how postcodes are managed within the system.

The core logic for postcode management is contained within the various subroutines in AF010R. The program handles user interactions through a series of subfiles, allowing users to create, update, delete, and view postcode entries.

Order Status and Header Rules

Postcode Management: aposl1, aposl2, aposlr, aposlu

1. Postcode Retrieval

- **Logic:** The program retrieves postcode information based on user input and displays it in a subfile.
- **File:** aposl1 (Postcode file)
- **Field:** aposl1_ponr
- **Condition:** The process will not select a record if b1valg is equal to 1, indicating a postcode retrieval request.

2. Postcode Update

- **Logic:** Allows users to update existing postcode records.
- **File:** aposlr (Postcode record)
- **Field:** aposlr_ponr
- **Condition:** The update occurs when b1valg is equal to 2, indicating an update request.

Configuration and Authorization Rules

1. User Authorization Check

- **Logic:** Validates user permissions before allowing changes to postcode records.
- **Files:**
- aposl1 (Postcode file)
- aposlu (Postcode update file)
- **Fields:**
- w_firm (Firm identifier from local data area)
- aposlu_ponr (Postcode to update)
- **Condition:** The program checks if the user has the necessary permissions based on the firm identifier.

2. Subfile Control Initialization

- **Logic:** Initializes the subfile control for displaying records.
- **File:** b2ctl (Subfile control)

- **Field:** *in14
- **Condition:** The control is set to active when the subfile is cleared.

Financial and Transactional Rules

1. Postcode Deletion

- **Logic:** Deletes a postcode record from the database.
- **File:** aposlu (Postcode update file)
- **Fields:**
- aposlu_ponr (Postcode to delete)
- w_firm (Firm identifier)
- **Condition:** The deletion occurs when b1valg is equal to 4, indicating a delete request.

2. Postcode Creation

- **Logic:** Creates a new postcode record in the database.
- **File:** aposlur (Postcode creation file)
- **Condition:** The creation occurs when b1valg is equal to 3, indicating a create request.

Special Conditions (Program-Specific)

1. Subfile Refresh (AF010R)

- **Logic:** Refreshes the subfile with the latest postcode data.
- **File:** b1sfl (Subfile for postcode records)
- **Field:** w_srrn
- **Condition:** The refresh occurs when the user navigates through the subfile, ensuring the data displayed is current. Note: This refresh logic is critical for maintaining data integrity during user interactions.

2. Postcode Display Logic (AF010R)

- **Logic:** Displays the details of a selected postcode for editing or viewing.
- **File:** aposlr (Postcode record)
- **Fields:** c2sted (Postcode description), c2fylk (Postal region)
- **Condition:** The display logic is triggered when a user selects a postcode to view or edit.

Subprogram Calls Affecting Logic

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

1. dsp_subfile (Display Subfile)

- **Trigger:** Called to display the current state of the subfile after any updates.
- **Logic:** This subroutine manages the display of the subfile and ensures that the user sees the most recent data.
- **Impact:** This call acts as a **major logical gateway** for user interactions with the postcode records.

2. crt_subfile (Create Subfile)

- **Trigger:** Invoked to populate the subfile with records from the database.
- **Logic:** This subroutine reads records from the postcode files and prepares them for display in the subfile.

- **Impact:** This is a **critical step** in ensuring that users have access to the latest postcode information.

3. xc2bld (Edit/Create Postcode Record)

- **Trigger:** Called when a user wants to create or edit a postcode record.
- **Logic:** This subroutine handles the logic for creating or updating postcode records based on user input.
- **Impact:** Represents the handoff to the next major business function, ensuring that changes to postcode records are processed correctly.