# AF010R.MBR

**Path**: NXCLOUD/rpgsrc/AF010R.MBR **Generated**: 2026-01-08 12:22:43 **Processing Time**: 12039ms

# 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 logic related to managing postcode entries, including creating, updating, deleting, and displaying records.
The core logic for maintaining postcode records is contained within the subfile subroutine in AF010R. The program processes records from the postcode files and manages user interactions through a series of subroutines designed for specific tasks.

## Order Status and Header Rules

Postcode Maintenance: aposl1, aposl2, aposlr, aposlu

**1. Record Retrieval**

•**Logic:** The program retrieves postcode records based on user input and displays them in a subfile format.
•**File:** aposl1 (Postcode file 1)
•**Field:** aposl1_ponr
•**Condition:** The process will not select a record if b2ponr is equal to 0 or b2sted is blank.

**2. Record Update**

•**Logic:** When a user selects a record for editing, the program updates the corresponding postcode entry in the database.
•**File:** aposlu (Postcode update file)
•**Field:** aposlu_ponr
•**Condition:** The update occurs if b1valg is equal to 2 (indicating an update action).

## Configuration and Authorization Rules

**1. User Permissions Check**

•**Logic:** The program checks user permissions before allowing modifications to postcode records.
•**Files:**
•aposl1 (Postcode file 1)
•aposlr (Postcode reference file)
•**Fields:**
•aposl1_ponr (Postcode number from file 1)
•aposlr_ponr (Postcode number from reference file)
•**Condition:** The program will only allow updates if the user has the necessary permissions, indicated by the variable b_forn.

**2. Record Deletion**

•**Logic:** The program allows users to delete a postcode record after confirmation.
•**File:** aposlu (Postcode update file)

•**Field:** aposlu_ponr
•**Condition:** The deletion occurs if b1valg is equal to 4 (indicating a delete action).

# Financial and Transactional Rules

**1. Record Creation**
•**Logic:** When a new postcode is created, the program inserts a new record into the postcode file.
•**File:** aposlr (Postcode reference file)
•**Fields:**
•aposlr_ponr (Postcode number)
•apsted (Postcode description)
•**Condition:** A new record is created if b1valg is equal to 1 (indicating a create action).
**2. Error Handling**
•**Logic:** The program displays error messages for invalid inputs or actions that cannot be performed.
•**File:** b1sfl (Subfile for error messages)
•**Condition:** Error messages are triggered based on user input validation checks.

# Special Conditions (Program-Specific)

**1. Postcode Retrieval (AF010R)**
•**Logic:** The program retrieves the postcode based on user selection and stores it in a variable for further processing.
•**File:** b1sfl (Subfile for postcode selection)
•**Field:** b1ponr
•**Condition:** This retrieval occurs if b1valg is equal to 1, indicating the user has selected a postcode.
**2. Display Message for Existing Record (AF010R)**
•**Logic:** If a user attempts to create a postcode that already exists, the program displays a message indicating the conflict.
•**File:** aposlr (Postcode reference file)
•**Fields:** aposlr_ponr (Postcode number), apsted (Postcode description)
•**Condition:** This check is performed when the user tries to create a postcode, and the program finds a matching entry.

# 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:** This subroutine is called to display the current state of the subfile after any changes.
•**Logic:** It checks if there are records to display and updates the display accordingly.
•**Impact:** This call ensures that the user interface reflects the most current data.
**2. clr_subfile (Clear Subfile)**
•**Trigger:** This subroutine is called to clear the current subfile before populating it with new data.

•**Logic:** It resets the subfile indicators and prepares it for new entries.
•**Impact:** This is a necessary step to avoid displaying stale data.

**3. crt_subfile (Create Subfile)**

•**Trigger:** This subroutine is called to populate the subfile with records from the postcode files.
•**Logic:** It reads records from the postcode files and writes them to the subfile for user interaction.
•**Impact:** This represents the main data loading step for the user interface, allowing users to interact with postcode records effectively.