AA000R.MBR

Path: NXCLOUD/rpgsrc/AA000R.MBR **Generated**: 2026-01-08 12:34:21 **Processing Time**: 10586ms

Business Logic for User Registration Lookup

This document outlines the business rules that govern the user registration lookup process, based on an analysis of the RPG program AA000R. The primary focus is on how the program retrieves and validates user information from the user register.

The core logic for user lookup is contained within the main processing section of the AA000R program. The program checks for the existence of a user record and validates the user against specific conditions before proceeding.

User Record Lookup Rules

User Registration Lookup: ausrl1

1. User Record Retrieval

- **Logic:** The program attempts to retrieve a user record based on the provided user identifier (p_user).
- **File:** ausrl1 (User register file)
- **Field:** ausrl1_user
- **Condition:** The program will not proceed if the user record is not found (*in60 = *off).

2. User Validation

- **Logic:** The program checks if the user is valid based on specific conditions. If the user is from a predefined list or starts with 'Q', the user is considered invalid.
- **File:** ausrl1 (User register file)
- **Field:** p_user
- **Condition:** The process will not select a record if p_user is equal to 'ASPKASSE', 'NORGROS', 'ASPRMI', or if the first character of p_user is 'Q'.

Initialization and Parameter Rules

1. Initialization of Parameters

- **Logic:** Upon program initiation, parameters are initialized to default values.
- **Files:** None (Initialization logic)
- **Fields:** p_okok, p_user, p_jobb
- **Condition:** The program sets p_okok to 0 to indicate an unsuccessful initialization.

2. User Job Association

- **Logic:** The job identifier (p_jobb) is stored in a local data structure upon successful user record retrieval.
- **File:** ausrl1 (User register file)
- **Field:** dswsid
- **Condition:** This field is populated only if a valid user record is found.

Special Conditions (Program-Specific)

1. User from Online Store (AA000R)

•**Logic:** If the user is identified as being from an online store, there are no requirements for the user to be defined in the user register.

•**File:** ausrl1 (User register file)

•**Field:** p_user

•**Condition:** This check is performed to allow certain users to bypass the validation process.

2. Invalid User Check (AA000R)

•**Logic:** The program checks if the user belongs to a restricted list and sets the status accordingly.

•**File:** ausrl1 (User register file)

•**Fields:** p_user (User identifier), w_alfa (First character of user identifier)

•**Condition:** The program will set p_okok to 0 if the user is found in the restricted list or if the first character of p_user is 'Q'.

Subprogram Calls Affecting Logic

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

1. *inzsr (Initialization Subroutine)

•**Trigger:** This subroutine is called at the beginning of the program.

•**Logic:** It initializes the parameters to default values.

•**Impact:** This ensures that the program starts with a clean state and sets the necessary flags for further processing.

2. chain (File Access Logic)

•**Trigger:** The chain operation is executed to retrieve the user record.

•**Logic:** It attempts to find the user record in the ausrl1 file based on the provided user identifier.

•**Impact:** This is a critical step that determines whether the program can proceed with valid user data or not.

3. eval (Data Assignment Logic)

•**Trigger:** Various eval statements are executed to assign values to local data fields.

•**Logic:** These statements populate local variables with data from the user record if found.

•**Impact:** This data assignment is essential for the program to function correctly and provide the necessary user information for subsequent processes.