AS100R.MBR

Path: NXCLOUD/rpgsrc/AS100R.MBR **Generated**: 2026-01-08 12:21:18 **Processing Time**: 11573ms

Business Logic for Number Series Calculation and Update

This document outlines the business rules that govern the calculation and updating of number series in the RPG program AS100R. The primary focus is on the logic that determines how numbers are retrieved, updated, and validated within the system.

The core logic for number series management is contained within the various subroutines in AS100R. The program processes requests to either retrieve the next available number, return a previously used number, or validate a manually entered number against defined ranges.

Order Status and Header Rules

Number Series Management: NUMMER-REGISTER

1. Retrieve Standard Number

- **Logic:** The program retrieves the next available number from the automatic series if the request code (p_kode) is '0'.
- **File:** anumlur (Number Register)
- **Field:** ansist
- **Condition:** The process will not select a record if the record is not found (*in66 = *off).

2. Return Last Used Number

- **Logic:** If the request code is '1', the last used number is returned to the automatic series.
- **File:** anumlur (Number Register)
- **Field:** ansist
- **Condition:** The process checks if the last used number matches the current number before returning it.

Configuration and Authorization Rules

1. Manual Number Validation

- **Logic:** The program checks if a manually entered number falls within the defined range.
- **Files:**
- anuml1r (Number Register)
- **Fields:**
- p_numm (Manual number)
- **Condition:** The process will set p_kode to '7' if the manual number is less than the starting number (annfom) or greater than the ending number (anntom).

2. Check for Number Series Existence

- **Logic:** If no number series is found, the program sets an error code.
- **File:** anuml1r (Number Register)
- **Field:** anuml1_key
- **Condition:** The process will set p_kode to '9' if no matching record is found.

Financial and Transactional Rules

1. Update Number Register

•**Logic:** The program updates the number register with the next available number.

•**File:** anumlur (Number Register)

•**Fields:**

•ansist (Last used number)

•**Condition:** The update occurs only if the record is found (*in66 = *off).

2. Wrap Around Logic for Number Series

•**Logic:** If the next number exceeds the defined maximum, it wraps around to the minimum.

•**File:** anumlur (Number Register)

•**Condition:** The process checks if p_numm exceeds anntom and if wrapping is enabled (anwrap = '1').

Special Conditions (Program-Specific)

1. Manual Number Handling (AS100R)

•**Logic:** The program handles manual number requests by checking if the number is valid and within the defined range.

•**File:** anuml1r (Number Register)

•**Field:** p_numm

•**Condition:** If the manual number is invalid, the program sets p_kode to '7' and resets p_numm to zero.

2. Return Last Used Number Logic (AS100R)

•**Logic:** The program checks if the last used number can be returned based on the current state.

•**File:** anumlur (Number Register)

•**Fields:** p_numm (Current number), ansist (Last used number)

•**Condition:** The program only returns the last used number if it matches the current number.

Subprogram Calls Affecting Logic

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

1. mannnr (Manual Number Check)

•**Trigger:** Called when validating a manually entered number.

•**Logic:** Checks if the manual number is within the defined range.

•**Impact:** This call ensures that only valid manual numbers are accepted, preventing errors in number assignment.

2. nyttnr (Retrieve Next Number)

•**Trigger:** Called when the request code is '0'.

•**Logic:** Calculates the next available number in the automatic series.

•**Impact:** This is a critical step for ensuring that the system provides a valid and sequential number.

3. gammnr (Return Last Used Number)

- **Trigger:** Called when the request code is '1'.
- **Logic:** Returns the last used number to the series if it matches the expected conditions.
- **Impact:** This step is essential for maintaining the integrity of the number series and preventing duplicates.