FA120R.MBR

Path: NXCLOUD/rpgsrc/FA120R.MBR **Generated**: 2026-01-08 12:34:33 **Processing Time**: 14159ms

Business Logic for Job Register Status Maintenance

This document outlines the business rules that govern the Job Register Status Maintenance process, based on an analysis of the RPG program FA120R. The primary focus is on how job status information is retrieved, displayed, and updated within the job register.

The core logic for maintaining job status is contained within the *inzsr subroutine in FA120R. The program processes job records by reading from the job register files, updating them based on user input, and managing user information associated with each job.

Job Status and Header Rules

Job Register: fjobl1, fjoblu, fausrl1

1. Retrieve Job Information

- **Logic:** The program retrieves job details from the job register file fjobl1. If the record is found, it populates various job-related fields; if not, it initializes them to zero.
- **File:** fjobl1 (Job Register File)
- **Field:** fajb01 to fajb10
- **Condition:** The process will not select a record if the chain operation to fjobl1 results in an error (indicated by *in90 being on).

2. Display Job Information

- **Logic:** After retrieving job information, the program displays the job details on the screen using the a1bld format.
- **File:** ffa120d (Display File)
- **Field:** Various job fields (a1jb01 to a1jb10)
- **Condition:** The display is executed unconditionally after job information is retrieved.

Configuration and Authorization Rules

1. User Lookup for Job Start

- **Logic:** The program checks which user started the job by looking up user information based on the job's field. It retrieves the user's name and other details if the corresponding field is active.
- **Files:**
- fausrl1 (User Register File)
- **Fields:**
- ausrl1_user (User ID)
- **Condition:** The lookup is performed based on the active field (w_afld), which determines which user field to check.

2. Update Job Register

- **Logic:** The program updates the job register with the current job information and timestamps when a job is modified.

•**File:** fjoblu (Job Update File)
•**Field:** fajb01 to fajb10
•**Condition:** The update occurs only if the record is found in fjoblu (indicated by *in90 being off).

Financial and Transactional Rules

1. Timestamping Job Updates

•**Logic:** When updating a job record, the program captures the current time for various fields to track when the job was last modified.
•**File:** fjoblu (Job Update File)
•**Fields:**
•fajeda (End Date)
•fajeti (End Time)
•**Condition:** Timestamps are only set if the job record is being updated (indicated by the absence of errors in the chain operation).

2. Job Creation Logic

•**Logic:** When a new job record is created, the program initializes fields with the current user and firm information.
•**File:** fjoblu (Job Update File)
•**Condition:** This logic is executed when a new job is being written to the database.

Special Conditions (Program-Specific)

1. Error Handling for User Lookup (FA120R)

•**Logic:** If the user lookup fails, the program does not display a user name and leaves it blank.
•**File:** fausrl1 (User Register File)
•**Field:** w_bnav (User Name)
•**Condition:** The user name is set to blank if the chain operation to fausrl1 results in an error (indicated by *in91 being on).

2. Job Record Initialization (FA120R)

•**Logic:** The program initializes keys for reading and updating job records at the start of the program.
•**File:** fjobl1, fjoblu (Job Register Files)
•**Fields:** w_firm (Firm Number)
•**Condition:** This initialization occurs unconditionally at the beginning of the program.

Subprogram Calls Affecting Logic

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

1. x_user (User Lookup Subroutine)

•**Trigger:** Called when the program needs to retrieve user information based on the job's user field.
•**Logic:** The subroutine looks up the user in the user register and sets the user name if found.
•**Impact:** This call ensures that the job records are associated with the correct user information.

2. *inzsr (Initialization Subroutine)

•**Trigger:** Called at the start of the program to set up keys for job record access.

•**Logic:** Initializes keys for reading and updating job records and sets the firm number.

•**Impact:** This is crucial for ensuring that the program can access the correct job records based on the firm.

3. exfmt (Display Format Subroutine)

•**Trigger:** Called to display job information on the screen.

•**Logic:** Displays the job details formatted according to the specified display file.

•**Impact:** This represents a key step in providing user feedback and interaction within the program.