VA551R.MBR

Type: Single File Analysis **Path**: NXCLOUD/rpgsrc/VA551R.MBR **Generated**: 2026-01-29 12:15:41 **Processing Time**: 18423ms

---

```
# Business Logic for VA551R

This document outlines the business rules that govern the processing of order types and accumulator type 2 inquiries, based on an analysis of the RPG program `VA551R`. The primary focus is on the conditions that will **block** an order type from being processed.

The core logic for handling subfile processing and order type validation is contained within the `crt_subfile` subroutine in `VA551R`. An order type must pass all the following checks to be included in the subfile and processed.

## Order Status and Header Rules

1.  **Order Type Must Be Valid**
    *   **Logic:** The order type (`vaotyp`) must not start with an asterisk (`*`) unless the second character matches the second character of the order type being processed.
    *   **File:** `VOTYL1` (Order Type File)
    *   **Field:** `VAOTYP`
    *   **Condition:** The process will skip an order type if `%subst(b2otyp:1:1) = '*' AND %subst(b2otyp:2:1) <> %subst(vaotyp:2:1)`. This ensures that only valid order types are included.

2.  **Order Type Must Be Pickable for Invoicing**
    *   **Logic:** The accumulator type (`vaoakk`) must be equal to `2` for the order type to be included in the subfile.
    *   **File:** `VOTYL1` (Order Type File)
    *   **Field:** `VAOAKK`
    *   **Condition:** The process will skip an order type if `VAOAKK <> 2`. This ensures that only order types that can be picked for invoicing are included.

## Configuration and Authorization Rules

3.  **Firm Code Must Match**
    *   **Logic:** The firm code (`vaofir`) of the order type must match the firm code passed as a parameter to the program.
    *   **File:** `VOTYL1` (Order Type File)
    *   **Field:** `VAOFIR`
    *   **Condition:** The process will skip an order type if `VAOFIR <> W_FIRM`. This ensures that only order types belonging to the specified firm are processed.

## Financial and Transactional Rules

4.  **Order Type Description Must Match (Optional)**
    *   **Logic:** If a description (`vaotxt`) is provided, the program will position the subfile based on the description and only include matching records.
    *   **File:** `VOTYL2` (Order Type Description File)
    *   **Field:** `VAOTXT`
    *   **Condition:** The process will position the subfile using the key `VOTYL2_KEY` if `B2OTXT <> *BLANK`. Orders not matching the description will not be included.

## Special Conditions (Program-Specific)

5.  **Subfile Initialization**
    *   **Logic:** The subfile is cleared and reloaded with records that meet all the above
```

conditions.
    *   **File:** `B1SFL` (Subfile for Order Types)
    *   **Field:** `B1OTYP`, `B1OTXT`
    *   **Condition:** The subfile is cleared using the `clr_subfile` subroutine and populated using the `crt_subfile` subroutine. Records that do not meet the conditions are skipped.

6. **Cursor Positioning**
    *   **Logic:** If specific positioning criteria are provided (order type or description), the program will position the subfile accordingly.
    *   **File:** `VOTYL1`, `VOTYL2` (Order Type Files)
    *   **Field:** `VAOTYP`, `VAOTXT`
    *   **Condition:** The program evaluates `B2OTYP` and `B2OTXT` for positioning. If both are blank, the subfile is populated from the beginning.

## Subprogram Calls Affecting Logic

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

1. **`clr_subfile` (Clear Subfile)**
    *   **Trigger:** Called before populating the subfile to ensure it is empty.
    *   **Logic:** Clears all subfile records and resets subfile counters.
    *   **Impact:** Ensures that only valid records are displayed in the subfile.

2. **`crt_subfile` (Create Subfile)**
    *   **Trigger:** Called during initialization and when the subfile needs to be populated.
    *   **Logic:** Reads records from the `VOTYL1` and `VOTYL2` files and writes them to the subfile if they meet all conditions.
    *   **Impact:** Populates the subfile with valid order types for processing.

3. **`posisjoner` (Positioning Subfile)**
    *   **Trigger:** Called when specific positioning criteria (order type or description) are provided.
    *   **Logic:** Positions the database cursor based on the provided criteria and reloads the subfile.
    *   **Impact:** Ensures that the subfile is positioned correctly for user interaction.

4. **`bck_subfile` (Back Subfile)**
    *   **Trigger:** Called when the user navigates backward in the subfile.
    *   **Logic:** Reads previous records and repopulates the subfile.
    *   **Impact:** Allows navigation to earlier records in the subfile.

5. **`dsp_subfile` (Display Subfile)**
    *   **Trigger:** Called to display the subfile to the user.
    *   **Logic:** Handles the display of the subfile and manages cursor positioning.
    *   **Impact:** Ensures that the subfile is displayed correctly to the user.