COBOL - Technical Design Specification for Modernization: EXWWB911

# 1. Introduction

## 1.1 Purpose

The purpose of program EXWWB911 is to extract vehicle and dealer data from various GEVIS-owned DB2 tables. This data is then used to create an outbound bridge file intended for Marketing Associates.

## 1.2 Scope

EXWWB911 is a batch program. It processes a list of producers from an input SYSPARM file one at a time. For each producer, it reads fleet sales data from MEXW001\_VEH\_ORDER and MEXW008\_VEH\_RTL tables that have been updated since the last program run. Only data associated with a current WDMO (Wholesale Dealer Marketing Organization) dealer is processed and written to the output bridge file. The program handles instances where DB2 fields might be unpopulated by outputting spaces for alphanumeric fields and zeros for numeric fields.

## 1.3 Audience

This document is intended for COBOL developers, system analysts, and testers involved in the maintenance, modernization, or understanding of the EXWWB911 program and its interfaces.

# 2. Overview

## 2.1 Background

EXWWB911 was originally created in November 2011 to provide Marketing Associates with vehicle and dealer data. It has undergone several revisions to modify its processing logic, such as removing a call to subroutine EXWWSWER, changing error handling for missing data in MEXW027\_CONV, updating SQL queries for MEXW034\_VL\_BRAND and MEXW032\_CATALOG, and refining criteria for MEXW003\_VEH\_STATUS to pull specific wholesale data. The output file FDEX.WW.PROD.EXWWB490.MKTASC.FLTSLS is picked up by Marketing Associates using FTP ID FDFTPMK.

## 2.2 Objectives

The primary objective of EXWWB911 is to: \* Read a list of producers from the SYSPARM file. \* For each producer, extract relevant vehicle and dealer data from GEVIS DB2 tables, focusing on fleet sales updated since the last run. \* Filter data to include only records associated with current WDMO dealers. \* Create an outbound bridge file formatted for Marketing Associates. \* Maintain run control information (timestamp, batch number) in MEXS016\_GENERIC2. \* Provide an audit trail of its processing.

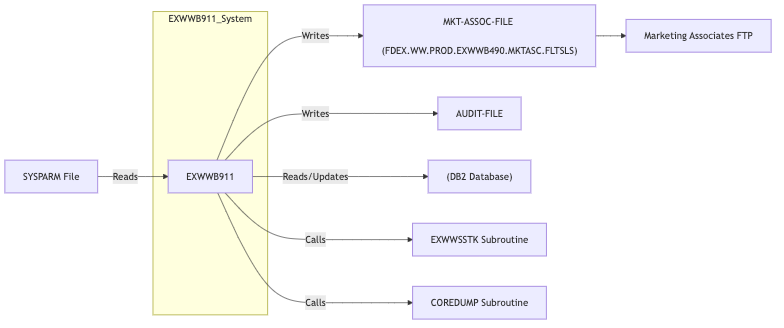
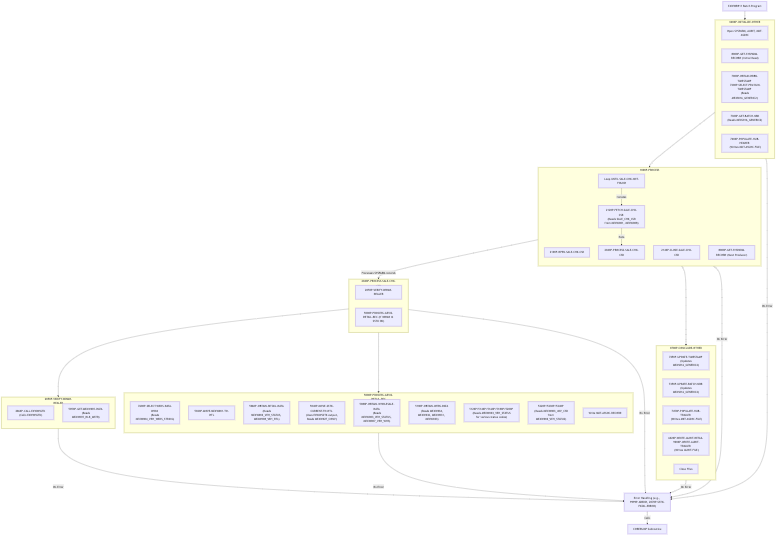
## 2.3 Assumptions and Constraints

* The input SYSPARM file (PARMFILE) contains a list of producers to be processed.
* DB2 tables (MEXW001\_VEH\_ORDER, MEXW008\_VEH\_RTL, MEXW003\_VEH\_STATUS, MEXW004\_VEH\_WERS\_STRING, MEXW007\_VEH\_WHS, MEXS016\_GENERIC2, MEXW021\_SUBLVL\_ASG, MEXW027\_CONV, MEXW031\_CATMAP, MEXW032\_CATALOG, MEXW033\_BODY\_TYPE, MEXW034\_VL\_BRAND, MEXW035\_DLR\_MSTR) are accessible and populated.
* The subroutine EXWWSSTK is available for determining current stocking dealer information.
* The subroutine COREDUMP is available for abend processing.
* The output file for Marketing Associates is FDEX.WW.PROD.EXWWB490.MKTASC.FLTSLS and is transferred via FTP using ID FDFTPMK.
* If a status row is not found on MEXW027\_CONV for source ‘VI’, the job will end with condition code 3, and an email will be sent. The transaction will not be written to the output file.
* Unpopulated alphanumeric fields in DB2 will be output as spaces; unpopulated numeric fields will be output as zeros.

# 3. System Architecture

## 3.1 System Context Diagram

### 3.1 System Context Diagram

 ### 3.2 Component Diagram 

# 4. Detailed Design  
## 4.1 Program Structure  
The program EXWWB911 follows a standard batch processing structure:  
1. \*\*Initialization Phase (`0400P-INITIALIZE-OTHER`)\*\*:  
 \* Opens input `SYSPARM-FILE`, output `MKT-ASSOC-FILE`, and extends `AUDIT-FILE`.  
 \* Initializes program switches, variables, and the output record layout.  
 \* Obtains the current Dearborn timestamp (`7000P-OBTAIN-DRBN-TIMESTAMP`).  
 \* Retrieves the previous run's timestamp and batch number from `MEXS016\_GENERIC2` (`7200P-SELECT-PRV-RUN-TIMESTAMP`, `7300P-GET-BATCH-NBR`).  
 \* Calculates the current batch number.  
 \* Writes audit headers (`9200P-WRITE-AUDIT-HEADER`).  
 \* Reads the first record from `SYSPARM-FILE` (`8000P-GET-SYSPARM-RECORD`).  
 \* Populates and writes the HUB header to `MKT-ASSOC-FILE` (`7400P-POPULATE-HUB-HEADER`).  
  
2. \*\*Main Processing Phase (`1000P-PROCESS`)\*\*:  
 \* This phase is performed in a loop for each producer read from `SYSPARM-FILE` until `END-OF-SYSPARM-FILE` is true.  
 \* Opens the main driving cursor `SALE\_CHK\_CSR` (`2100P-OPEN-SALE-CHK-CSR`). This cursor joins `MEXW001\_VEH\_ORDER` and `MEXW008\_VEH\_RTL` for fleet sales updated since the last run for the current producer.  
 \* Fetches records from `SALE\_CHK\_CSR` (`2120P-FETCH-SALE-CHK-CSR`).  
 \* For each fetched record, it processes the data via `2040P-PROCESS-SALE-CHK-CSR`:  
 \* Verifies if the dealer is WDMO based by calling subroutine `EXWWSSTK` to get the current stocking dealer, then checks `MEXW035\_DLR\_MSTR` (`2050P-VERIFY-WDMO-DEALER`).  
 \* If it's a WDMO dealer and `EXWWSSTK` call was successful, it performs `5000P-PROCESS-GEVIS-DETAIL-REC`.  
 \* This involves gathering additional data by selecting from various DB2 tables (`MEXW004`, `MEXW003`, `MEXW008`, `MEXW007`, `MEXW027`, `MEXW031`, `MEXW032`, `MEXW033`, `MEXW034`) and using cursors `MEXW031\_CSR` and `MEXW003\_40V\_CSR`.  
 \* Formats and moves data to `WS-MKT-ASSOC-DETAIL-RECORD`.  
 \* Writes the record to `MKT-ASSOC-FILE`.  
 \* Closes `SALE\_CHK\_CSR` (`2140P-CLOSE-SALE-CHK-CSR`).  
 \* Reads the next producer record from `SYSPARM-FILE` (`8000P-GET-SYSPARM-RECORD`).  
  
3. \*\*Conclusion Phase (`0700P-CONCLUDE-OTHER`)\*\*:  
 \* Updates the run timestamp and batch number in `MEXS016\_GENERIC2` (`7250P-UPDATE-TIMESTAMP`, `7350P-UPDATE-BATCH-NBR`).  
 \* Populates and writes the HUB trailer to `MKT-ASSOC-FILE` (`7450P-POPULATE-HUB-TRAILER`).  
 \* Writes audit detail and trailer records to `AUDIT-FILE` (`6020P-WRITE-AUDIT-DETAIL`, `9300P-WRITE-AUDIT-TRAILER`).  
 \* Closes all files.  
 \* Sets `RETURN-CODE` to 3 if `SEND-EMAIL` switch is 'Y' (e.g., due to missing `MEXW027` row or `EXWWSSTK` +100 error).  
  
Error handling is performed throughout the program. SQL errors typically lead to `9999P-ABEND`, which calls `COREDUMP`. Specific conditions like missing `MEXW027` data or `EXWWSSTK` errors may set flags to send an email and set a specific return code.  
  
## 4.2 Data Structures  
  
### File Record Layouts:  
  
\* \*\*AUDIT-RECORD\*\* (Output File `AUDIT-FILE`): Used for writing audit trail information.  
 ```COBOL  
 FD AUDIT-FILE.  
 01 AUDIT-RECORD.  
 05 AUDIT-LABEL PIC X(30).  
 05 AUDIT-DATA PIC X(50).  
 ```  
 \* Copybooks referenced: None for this specific FD, but `CPESEBWS` is used for audit message formatting.  
  
\* \*\*SYSPARM-RECORD\*\* (Input File `SYSPARM-FILE`): Contains producer information to be processed.  
 ```COBOL  
 FD SYSPARM-FILE.  
 01 SYSPARM-RECORD PIC X(80).  
 ```  
 The program uses `WS-SYSPARM-DETAIL` to structure this input:  
 ```COBOL  
 01 WS-SYSPARM-DETAIL.  
 05 WS-SYSPARM-DATA-SRC PIC X(02).  
 05 FILLER PIC X(78).  
 ```  
 \* Copybooks referenced: None.  
  
\* \*\*MKT-ASSOC-RECORD\*\* (Output File `MKT-ASSOC-FILE`): The main output bridge file for Marketing Associates.  
 ```COBOL  
 FD MKT-ASSOC-FILE.  
 01 MKT-ASSOC-RECORD PIC X(1000).  
 ```  
 The actual data is structured using `WS-MKT-ASSOC-DETAIL-RECORD` and also HUB records from `CPEWHUB`.  
 `WS-MKT-ASSOC-DETAIL-RECORD` hierarchical outline:  
 ```  
 01 WS-MKT-ASSOC-DETAIL-RECORD.  
 05 WS-DTL-HUB-LINE-NBR PIC 9(06).  
 05 WS-DTL-HUB-REC-ID PIC X(25).  
 05 WS-DTL-HUB-REC-ID-NBR REDEFINES WS-DTL-HUB-REC-ID.  
 10 WS-DTL-REC-ID-ZEROS PIC 9(16).  
 10 WS-DTL-REC-ID-NBR PIC 9(09).  
 05 WS-DTL-HUB-REC-TYPE PIC X(03).  
 05 WS-DTL-HUB-REC-SEQ-NBR PIC 9(03).  
 05 WS-DTL-VIN-FULL-C PIC X(17).  
 05 WS-DTL-DTA-DATA-SRC-C PIC X(02).  
 05 WS-DTL-BDT-MDL-YR-Y PIC X(02).  
 05 WS-DTL-GEVIS-VEH-LINE-C PIC X(02).  
 05 WS-DTL-LCL-BDYTYP-C PIC X(05).  
 05 WS-DTL-CUR-STA-STATUS-C PIC X(03).  
 05 WS-DTL-VEH-DIVISION-C PIC X(01).  
 05 WS-DTL-WMI-WMI-C PIC X(03).  
 05 WS-DTL-LCL-PLT-C PIC X(03).  
 05 WS-DTL-VWS-TOT-US-A PIC S9(07)V99 COMP-3.  
 05 WS-DTL-VEH-GBL-DLR-C PIC X(06).  
 05 WS-DTL-ORIG-VST-GBL-LOC-C PIC X(06).  
 05 WS-DTL-CURR-VST-GBL-LOC-C PIC X(06).  
 05 WS-DTL-SHIP-TO-DLR-C PIC X(06).  
 05 WS-DTL-CURR-STOCKING-DLR-C PIC X(06).  
 05 WS-DTL-CURR-DLR-C PIC X(06).  
 05 WS-DTL-WDMO-FLEET-C PIC X(05).  
 05 WS-DTL-VRS-LCL-FLEET-C PIC X(06).  
 05 WS-DTL-VRS-CST-FIRST-N PIC X(30).  
 05 WS-DTL-VRS-CST-MID-INIT-X PIC X(01).  
 05 WS-DTL-VRS-CST-LAST-N PIC X(30).  
 05 WS-DTL-VRS-CST-ADDR-1-X PIC X(40).  
 05 WS-DTL-VRS-CST-ADD-DIV2-N PIC X(40).  
 05 WS-DTL-VRS-CST-ADD-DIV1-C PIC X(02).  
 05 WS-DTL-VRS-CST-POSTAL-C PIC X(10).  
 05 WS-DTL-VRS-SALESPERSON-C PIC X(11).  
 05 WS-DTL-VRS-TYP-LCL-CUST-C PIC X(01).  
 05 WS-DTL-VEH-WDMO-ORD-TYP PIC X(01).  
 05 WS-DTL-VEH-ORD-RCPT-Y PIC X(08).  
 05 WS-DTL-VEH-SCHD-VST-TARGET-Y PIC X(08).  
 05 WS-DTL-VEH-PRODUCE-VST-STAT-Y PIC X(08).  
 05 WS-DTL-VEH-RELEASE-VST-STAT-Y PIC X(08).  
 05 WS-DTL-VEH-ARRIVAL-VST-STAT-Y PIC X(08).  
 05 WS-DTL-VEH-INVOICE-VST-STAT-Y PIC X(08).  
 05 WS-DTL-VEH-STOCK-VST-STAT-Y PIC X(08).  
 05 WS-DTL-VEH-RETAIL-VST-STAT-Y PIC X(08).  
 05 WS-DTL-VEH-DELIVER-VST-STAT-Y PIC X(08).  
 05 WS-DTL-VEH-SLSRCPT-VST-STAT-Y PIC X(08).  
 05 WS-DTL-VEH-WARRANT-VST-STAT-Y PIC X(08).  
 05 WS-DTL-VEH-CATALOG-C PIC X(15).  
 05 WS-DTL-WERS-VEH-LINE-C PIC X(02).  
 05 WS-DTL-WERS-BODY-STYLE-C PIC X(03).  
 05 WS-DTL-WERS-BRAND-C PIC X(01).  
 05 WS-DTL-VEH-PO-Y PIC X(08).  
 05 WS-DTL-VLN-GEVIS-VL-N PIC X(40).  
 05 WS-DTL-FILLER-01 PIC X(550).  
 ```  
 \* Copybooks referenced: `CPEWHUB` for HUB header/trailer records written to this file.  
  
### Called Program Interface Structures:  
  
\* \*\*SSTK-I-O-DATA\*\* (for `EXWWSSTK` Subroutine): Used to pass input parameters to and receive output data from the `EXWWSSTK` subroutine.  
 Hierarchical outline:  
 ```  
 01 SSTK-I-O-DATA.  
 05 SSTK-INPUT-DATA.  
 10 SSTK-MODE PIC X(01).  
 88 SSTK-INQUIRY-MODE VALUE "I".  
 88 SSTK-UPDATE-MODE VALUE "U".  
 10 SSTK-DTA-DATA-SRC-C PIC X(02).  
 10 SSTK-VEH-ORD-ID-C PIC X(25).  
 05 SSTK-OUTPUT-DATA.  
 10 SSTK-GBL-STK-DLR-C PIC X(06).  
 10 SSTK-LCL-STK-DLR-C PIC X(07).  
 10 SSTK-STK-DLR-STAT-C PIC X(03).  
 10 SSTK-STK-DLR-STAT-Y PIC X(10).  
 10 SSTK-STK-DLR-CNTRY-ISO3-C PIC X(03).  
 10 SSTK-CUR-STAT-C PIC X(03).  
 10 SSTK-CUR-LCL-STAT-C PIC X(06).  
 10 SSTK-CUR-STAT-Y PIC X(10).  
 10 SSTK-DIV-DIV-C PIC X(02).  
 05 SSTK-OUT-DATA-MSG.  
 10 SSTK-PGM-ID PIC X(08).  
 10 SSTK-RETURN-CD PIC X(01).  
 88 SSTK-SUCCESSFUL VALUE "0".  
 88 SSTK-INPUT-ERROR VALUE "1".  
 88 SSTK-DB2-ERROR VALUE "2".  
 10 SSTK-PARAGRAPH PIC X(06).  
 10 SSTK-DB2-AREA.  
 15 SSTK-HOST-VAR1 PIC X(80).  
 15 SSTK-HOST-VAR2 PIC X(80).  
 15 SSTK-HOST-VAR3 PIC X(80).  
 15 SSTK-HOST-VAR4 PIC X(80).  
 15 SSTK-HOST-VAR5 PIC X(80).  
 15 SSTK-HOST-VAR6 PIC X(80).  
 15 SSTK-HOST-VAR7 PIC X(80).  
 15 SSTK-HOST-VAR8 PIC X(80).  
 15 SSTK-DB2-TABLES.  
 20 SSTK-DB2-TABLE1 PIC X(18).  
 20 SSTK-DB2-TABLE2 PIC X(18).  
 20 SSTK-DB2-TABLE3 PIC X(18).  
 20 SSTK-DB2-TABLE4 PIC X(18).  
 20 SSTK-DB2-TABLE5 PIC X(18).  
 15 SSTK-SQL-FUNCTION PIC X(12).  
 15 SSTK-SQL-RETURN-CODE PIC S9(04) COMP-3.  
 15 SSTK-SQL-WARNING PIC X(08).  
 15 SSTK-SQL-ERROR-MESSAGE PIC X(70).  
 15 SSTK-SQL-FULL-ERROR.  
 20 SSTK-SQL-MSG1 PIC X(72).  
 20 SSTK-SQL-MSG2 PIC X(72).  
 20 SSTK-SQL-MSG3 PIC X(72).  
 20 SSTK-SQL-MSG4 PIC X(72).  
 15 SSTK-SQLCA PIC X(200).  
 05 SSTK-FILLER PIC X(1596).  
 ```  
 \* Copybooks referenced: `CPEWSSTK`.  
  
### Other Significant Working-Storage Structures:  
The program uses numerous DCLGEN copybooks for DB2 table layouts (e.g., `CPEWD001` for `MEXW001-VEH-ORDER`, `CPEWD003` for `MEXW003-VEH-STATUS`, etc.). These define the host variables used in SQL statements.  
  
## 4.3 Algorithms  
### 4.3.1 Overall Program Logic (Condensed Pseudocode)

START PROGRAM EXWWB911

PERFORM 0400P-INITIALIZE-OTHER Open SYSPARM-FILE, AUDIT-FILE, MKT-ASSOC-FILE Initialize switches and variables PERFORM 7000P-OBTAIN-DRBN-TIMESTAMP (DB2 CURRENT TIMESTAMP) PERFORM 7200P-SELECT-PRV-RUN-TIMESTAMP (DB2 SELECT from MEXS016\_GENERIC2 for previous timestamp) PERFORM 7300P-GET-BATCH-NBR (DB2 SELECT from MEXS016\_GENERIC2 for previous batch number) Calculate PV-CURRENT-BATCH-NBR PERFORM 9100P-GET-CURRENT-DATE-TIME PERFORM 9200P-WRITE-AUDIT-HEADER PERFORM 8000P-GET-SYSPARM-RECORD (Read first SYSPARM record) PERFORM 7400P-POPULATE-HUB-HEADER (Write MKT-ASSOC-FILE header)

PERFORM 1000P-PROCESS UNTIL END-OF-SYSPARM-FILE PERFORM 2100P-OPEN-SALE-CHK-CSR (Open main DB2 cursor) PERFORM 2120P-FETCH-SALE-CHK-CSR (Fetch first record from cursor) PERFORM 2040P-PROCESS-SALE-CHK-CSR UNTIL SALE-CHK-NOT-FOUND PERFORM 2050P-VERIFY-WDMO-DEALER PERFORM 2060P-CALL-EXWWSSTK (Call EXWWSSTK subroutine) IF SSTK-SUCCESSFUL THEN Use SSTK-GBL-STK-DLR-C PERFORM 5050P-GET-MEXW035-DATA (DB2 SELECT from MEXW035\_DLR\_MSTR) IF MEXW035-FOUND AND DLR-SUB-SUBLVL1-C = ‘WDM’ THEN SET WDMO-DEALER TO TRUE ELSE SET NOT-WDMO-DEALER TO TRUE END-IF END-IF IF WDMO-DEALER AND SSTK-SUCCESSFUL THEN PERFORM 5000P-PROCESS-GEVIS-DETAIL-REC (Gather data from various DB2 tables: MEXW004, MEXW003, MEXW008, MEXW007, MEXW027, MEXW031, MEXW032, MEXW033, MEXW034) (Populate WS-MKT-ASSOC-DETAIL-RECORD) (Write MKT-ASSOC-RECORD) END-IF PERFORM 2120P-FETCH-SALE-CHK-CSR (Fetch next record from cursor) END-PERFORM PERFORM 2140P-CLOSE-SALE-CHK-CSR (Close main DB2 cursor) PERFORM 8000P-GET-SYSPARM-RECORD (Read next SYSPARM record) END-PERFORM

PERFORM 0700P-CONCLUDE-OTHER PERFORM 7250P-UPDATE-TIMESTAMP (DB2 UPDATE MEXS016\_GENERIC2 with current timestamp) PERFORM 7350P-UPDATE-BATCH-NBR (DB2 UPDATE MEXS016\_GENERIC2 with current batch number) PERFORM 7450P-POPULATE-HUB-TRAILER (Write MKT-ASSOC-FILE trailer) PERFORM 6020P-WRITE-AUDIT-DETAIL PERFORM 9100P-GET-CURRENT-DATE-TIME PERFORM 9300P-WRITE-AUDIT-TRAILER Close SYSPARM-FILE, AUDIT-FILE, MKT-ASSOC-FILE

IF SEND-EMAIL THEN SET RETURN-CODE TO 3 END-IF GOBACK END PROGRAM EXWWB911 ```

### 4.3.2 Key Algorithmic Details

* **Producer Processing**: The program iterates through producer codes supplied in the SYSPARM-FILE. For each producer, it re-opens and processes the SALE\_CHK\_CSR.
* **Main Data Extraction (SALE\_CHK\_CSR)**: This cursor selects fleet sales data from MEXW001\_VEH\_ORDER (aliased as VEH) and MEXW008\_VEH\_RTL (aliased as VRS). It filters by VRS.DTA\_DATA\_SRC\_C (from SYSPARM), VRS.VRS\_UPDT\_S (greater than previous run timestamp), VRS.VRS\_ACTIVE\_F (‘Y’), and specific VRS.VRS\_TYP\_LCL\_CUST\_C values. It also ensures VEH.VEH\_VIN\_FULL\_C is not blank and VEH.VEH\_ACTIVE\_F is ‘Y’.
* **WDMO Dealer Verification (2050P-VERIFY-WDMO-DEALER)**:
  1. Calls EXWWSSTK (2060P-CALL-EXWWSSTK) with the vehicle order ID and data source to get current stocking dealer (SSTK-GBL-STK-DLR-C).
  2. If EXWWSSTK is successful, it uses SSTK-GBL-STK-DLR-C to query MEXW035\_DLR\_MSTR (5050P-GET-MEXW035-DATA).
  3. If a record is found in MEXW035\_DLR\_MSTR and DLR-SUB-SUBLVL1-C is ‘WDM’, the dealer is considered a WDMO dealer.
* **GEVIS Detail Record Processing (5000P-PROCESS-GEVIS-DETAIL-REC)**:
  1. Initializes WERS string data and attempts to select WERS data from MEXW004\_VEH\_WERS\_STRING (7600P-SELECT-WERS-DATA-W004).
  2. Moves data from the SALE\_CHK\_CSR (from MEXW001\_VEH\_ORDER fields) to the output record (5020P-MOVE-MEXW001-TO-DTL). This includes deriving body style from WERS string if available, or from MEXW031\_CATMAP or VEH-LCL-BDYTYP-C.
  3. Obtains retail data from MEXW003\_VEH\_STATUS and MEXW008\_VEH\_RTL for ‘90V’ status (5060P-OBTAIN-RETAIL-DATA).
  4. Moves current stocking dealer info (from EXWWSSTK output) and converts status code using MEXW027\_CONV (5040P-MOVE-SSTK-CURRENT-TO-DTL). Handles special logic for status ‘800’ vs ‘801’ or ‘600’.
  5. Obtains wholesale data from MEXW003\_VEH\_STATUS and MEXW007\_VEH\_WHS for ‘40V’ status (5080P-OBTAIN-WHOLESALE-DATA).
  6. Obtains detailed WERS data (5100P-OBTAIN-WERS-DATA) by conditionally selecting from MEXW034\_VL\_BRAND, MEXW032\_CATALOG, MEXW031\_CATMAP, and MEXW033\_BODY\_TYPE based on data source and availability of WERS string from MEXW004.
  7. Obtains various vehicle status dates (Scheduled Target ‘20T’, Produced ‘30R’ or ‘30P’, Released ‘30T’, Arrived ‘80F’) by selecting from MEXW003\_VEH\_STATUS (5120P, 5140P, 5160P, 5180P, 5200P).
  8. Obtains original invoicing dealer using MEXW003\_40V\_CSR (5220P to 5240P).
  9. If MEXW027\_CONV lookup was successful, increments counters, populates HUB record IDs, and writes WS-MKT-ASSOC-DETAIL-RECORD to MKT-ASSOC-FILE.
* **Timestamp and Batch Number Management**:
  + At startup, reads previous run timestamp and batch number from MEXS016\_GENERIC2.
  + At conclusion, updates MEXS016\_GENERIC2 with the current run’s start timestamp and the incremented batch number.
* **WERS Body Style Derivation (5020P, 5100P)**:
  + If WERS string found (WERS-STRING-FOUND-SW from 7600P): WS-DTL-LCL-BDYTYP-C = VWR-WERS-STRING-X-TEXT(10:2). WS-DTL-WERS-BODY-STYLE-C = VWR-WERS-PRD-TP-C + VWR-WERS-STRING-X-TEXT(10:2).
  + Else (No WERS string):
    - Open/Fetch/Close MEXW031\_CSR (5300P to 5340P).
    - If MEXW031\_FOUND: WS-DTL-LCL-BDYTYP-C = CTM-OPT-OPTION-C. WS-DTL-WERS-BODY-STYLE-C = CTM-VPT-PROD-TYP-C + CTM-OPT-OPTION-C.
    - Else (MEXW031\_NOT\_FOUND): WS-DTL-LCL-BDYTYP-C = VEH-LCL-BDYTYP-C.
      * If data source is ‘EA’ or ‘NA’: Perform 5115P-SELECT-MEXW033-DATA. If found and BDT-WERS-BDY-TYP-C > SPACES: WS-DTL-WERS-BODY-STYLE-C = VLN-WERS-PRD-TP-C (from MEXW034 via 5110P) + BDT-WERS-BDY-TYP-C.

## 4.4 Input/Output Specifications

* **Input Files**:
  + SYSPARM-FILE (Logical name: PARMFILE): Sequential file containing producer codes (data source codes) to process. Record layout: SYSPARM-RECORD PIC X(80), structured by WS-SYSPARM-DETAIL.
* **Output Files**:
  + MKT-ASSOC-FILE (Logical name: MKTASSOC): Sequential file, the primary output bridge file. Record layout: MKT-ASSOC-RECORD PIC X(1000). Contains HUB header, detail records from WS-MKT-ASSOC-DETAIL-RECORD, and HUB trailer. Dataset name: FDEX.WW.PROD.EXWWB490.MKTASC.FLTSLS.
  + AUDIT-FILE (Logical name: AUDIT): Sequential file for audit trail messages. Record layout: AUDIT-RECORD (label PIC X(30), data PIC X(50)).
* **External Interfaces**:
  + DB2 Tables: See section 4.5.
  + Called Subroutines: See section 4.7.
  + FTP: The MKT-ASSOC-FILE is picked up by Marketing Associates using FTP ID FDFTPMK.

## 4.5 DB2 Database Details

### Cursors:

1. **SALE\_CHK\_CSR**: Main driving cursor to select fleet sales data.

* EXEC SQL  
  DECLARE SALE\_CHK\_CSR CURSOR FOR  
  SELECT VEH.VEH\_VIN\_FULL\_C  
   ,VEH.VEH\_ORD\_ID\_C  
   ,VEH.DTA\_DATA\_SRC\_C  
   ,VEH.BDT\_MDL\_YR\_Y  
   ,VEH.WMI\_WMI\_C  
   ,VEH.VEH\_LCL\_PLT\_C  
   ,VEH.VEH\_LCL\_BDYTYP\_C  
   ,VEH.VEH\_GBL\_ORD\_DLR\_C  
   ,VEH.VEH\_GBL\_SHIP\_TO\_C  
   ,VEH.VEH\_ORD\_RCPT\_Y  
   ,VEH.VEH\_WDMO\_FLEET\_C  
   ,VEH.VEH\_WDMO\_ORD\_TYP  
   ,VEH.VEH\_CATALOG\_C  
   ,VEH.VEH\_GBL\_CATALOG\_C  
   ,VEH.VEH\_PO\_Y  
   ,VEH.VEH\_GEVIS\_VL\_C  
   ,VEH.COUNTRY\_ISO3\_C  
  FROM MEXW001\_VEH\_ORDER VEH  
   ,MEXW008\_VEH\_RTL VRS  
  WHERE VRS.DTA\_DATA\_SRC\_C = :VRS-DTA-DATA-SRC-C  
   AND VRS.VRS\_UPDT\_S > :PV-PREV-RUN-TIMESTAMP  
   AND VRS.VRS\_ACTIVE\_F = :PL-Y  
   AND VRS.VRS\_TYP\_LCL\_CUST\_C IN (:PL-F1,  
   :PL-7,  
   :PL-F2,  
   :PL-3,  
   :PL-F3,  
   :PL-T,  
   :PL-F4,  
   :PL-4,  
   :PL-L2,  
   :PL-L4,  
   :PL-D)  
   AND VEH.VEH\_ORD\_ID\_C = VRS.VEH\_ORD\_ID\_C  
   AND VEH.DTA\_DATA\_SRC\_C = VRS.DTA\_DATA\_SRC\_C  
   AND VEH.VEH\_VIN\_FULL\_C > :PL-VEH-VIN-BLANK  
   AND VEH.VEH\_ACTIVE\_F = :PL-Y  
  END-EXEC
  + Tables Referenced: MEXW001\_VEH\_ORDER, MEXW008\_VEH\_RTL

1. **MEXW031\_CSR**: Selects option and product type from MEXW031\_CATMAP.

* EXEC SQL  
  DECLARE MEXW031\_CSR CURSOR FOR  
  SELECT OPT\_OPTION\_C  
   ,VPT\_PROD\_TYP\_C  
  FROM MEXW031\_CATMAP  
  WHERE DTA\_DATA\_SRC\_C = :CTM-DTA-DATA-SRC-C  
   AND CTM\_LCL\_CATALOG\_C = :CTM-LCL-CATALOG-C  
   AND OFM\_OPTION\_FAM\_C IN (:PL-BS, :PL-CA)  
  OPTIMIZE FOR 1 ROW  
  END-EXEC
  + Tables Referenced: MEXW031\_CATMAP

1. **MEXW003\_40V\_CSR**: Retrieves original invoicing dealer from MEXW003\_VEH\_STATUS.

* EXEC SQL  
  DECLARE MEXW003\_40V\_CSR CURSOR FOR  
  SELECT VST\_GBL\_LOC\_C  
  FROM MEXW003\_VEH\_STATUS  
  WHERE VEH\_ORD\_ID\_C = :VST-VEH-ORD-ID-C  
   AND DTA\_DATA\_SRC\_C = :VST-DTA-DATA-SRC-C  
   AND STA\_STATUS\_C = :VST-STA-STATUS-C -- Expected :PL-40V  
   AND VST\_ACTIVE\_F = :VST-ACTIVE-F -- Expected :PL-Y  
   AND VST\_STAT\_TYP\_C = :VST-STAT-TYP-C -- Expected :PL-D  
   AND VST\_CUR\_DATA\_SRC\_C = :VST-CUR-DATA-SRC-C -- Expected :PL-WD  
  ORDER BY VST\_STAT\_Y DESC  
   ,VST\_STATIC\_ISRT\_REC\_S DESC  
  END-EXEC
  + Tables Referenced: MEXW003\_VEH\_STATUS

### Singleton SQL Statements:

1. **Get Current Timestamp (7000P-OBTAIN-DRBN-TIMESTAMP)**:

* EXEC SQL  
  SET :PV-CURR-DRBN-TIMESTAMP = CURRENT TIMESTAMP  
  END-EXEC
  + Tables Referenced: None (DB2 special register)

1. **Select Previous Run Timestamp (7200P-SELECT-PRV-RUN-TIMESTAMP)**:

* EXEC SQL  
  SELECT GNT\_ATTRIBUTE\_DATA  
  INTO :GNT-ATTRIBUTE-DATA  
  FROM MEXS016\_GENERIC2  
  WHERE GNT\_SYSTEM\_CD = :GNT-SYSTEM-CD -- Input: PL-SYSTEM-ID  
  AND GNT\_TABLE\_ID = :GNT-TABLE-ID -- Input: PL-TABLE-ID  
  AND GNT\_KEY\_DATA = :GNT-KEY-DATA -- Input: PL-KEY-DATA-TIMESTAMP  
  AND GNT\_SW\_ACTIVE LIKE :GNT-SW-ACTIVE -- Input: PL-Y  
  END-EXEC
  + Tables Referenced: MEXS016\_GENERIC2

1. **Update Run Timestamp (7250P-UPDATE-TIMESTAMP)**:

* EXEC SQL  
  UPDATE MEXS016\_GENERIC2  
  SET GNT\_ATTRIBUTE\_DATA = :GNT-ATTRIBUTE-DATA -- Input: PV-CURR-DRBN-TIMESTAMP  
  WHERE GNT\_SYSTEM\_CD = :GNT-SYSTEM-CD -- Input: PL-SYSTEM-ID  
  AND GNT\_TABLE\_ID = :GNT-TABLE-ID -- Input: PL-TABLE-ID  
  AND GNT\_KEY\_DATA = :GNT-KEY-DATA -- Input: PL-KEY-DATA-TIMESTAMP  
  END-EXEC
  + Tables Referenced: MEXS016\_GENERIC2

1. **Select Previous Batch Number (7300P-GET-BATCH-NBR)**:

* EXEC SQL  
  SELECT GNT\_ATTRIBUTE\_DATA  
  INTO :GNT-ATTRIBUTE-DATA  
  FROM MEXS016\_GENERIC2  
  WHERE GNT\_SYSTEM\_CD = :GNT-SYSTEM-CD -- Input: PL-SYSTEM-ID  
  AND GNT\_TABLE\_ID = :GNT-TABLE-ID -- Input: PL-TABLE-ID  
  AND GNT\_KEY\_DATA = :GNT-KEY-DATA -- Input: PL-KEY-DATA-BATCHCTL  
  AND GNT\_SW\_ACTIVE LIKE :GNT-SW-ACTIVE -- Input: PL-Y  
  END-EXEC
  + Tables Referenced: MEXS016\_GENERIC2

1. **Update Batch Number (7350P-UPDATE-BATCH-NBR)**:

* EXEC SQL  
  UPDATE MEXS016\_GENERIC2  
  SET GNT\_ATTRIBUTE\_DATA = :GNT-ATTRIBUTE-DATA -- Input: PV-CURRENT-BATCH-NBR  
  WHERE GNT\_SYSTEM\_CD = :GNT-SYSTEM-CD -- Input: PL-SYSTEM-ID  
  AND GNT\_TABLE\_ID = :GNT-TABLE-ID -- Input: PL-TABLE-ID  
  AND GNT\_KEY\_DATA = :GNT-KEY-DATA -- Input: PL-KEY-DATA-BATCHCTL  
  END-EXEC
  + Tables Referenced: MEXS016\_GENERIC2

1. **Select WERS Data from MEXW004 (7600P-SELECT-WERS-DATA-W004)**:

* EXEC SQL  
  SELECT VWR\_WERS\_STRING\_X  
   ,VWR\_WERS\_VL\_C  
   ,VWR\_WERS\_PRD\_TP\_C  
   ,VWR\_MAJ\_FEAT\_DFNED\_F  
  INTO :VWR-WERS-STRING-X  
   ,:VWR-WERS-VL-C  
   ,:VWR-WERS-PRD-TP-C  
   ,:VWR-MAJ-FEAT-DFNED-F  
  FROM MEXW004\_VEH\_WERS\_STRING  
  WHERE VEH\_ORD\_ID\_C = :VWR-VEH-ORD-ID-C  
   AND DTA\_DATA\_SRC\_C = :VWR-DTA-DATA-SRC-C  
  END-EXEC
  + Tables Referenced: MEXW004\_VEH\_WERS\_STRING

1. **Select from MEXW027\_CONV (5045P-SELECT-MEXW027-DATA)**:

* EXEC SQL  
  SELECT CNT\_LCL\_DATA\_X  
  INTO :CNT-LCL-DATA-X  
  FROM MEXW027\_CONV  
  WHERE CND\_CNV\_TYP\_C = :CNT-CND-CNV-TYP-C -- Input: PL-STATUS  
   AND DTA\_DATA\_SRC\_C = :CNT-DTA-DATA-SRC-C -- Input: PL-VI  
   AND CNT\_GBL\_DATA\_X = :CNT-GBL-DATA-X -- Input: SSTK-CUR-STAT-C  
  END-EXEC
  + Tables Referenced: MEXW027\_CONV

1. **Select from MEXW035\_DLR\_MSTR (5050P-GET-MEXW035-DATA)**:

* EXEC SQL  
  SELECT SUB\_SUBLVL1\_C  
   ,DLR\_SUPER\_DLR\_C  
  INTO :DLR-SUB-SUBLVL1-C  
   ,:DLR-SUPER-DLR-C  
  FROM MEXW035\_DLR\_MSTR  
  WHERE DLR\_DLR\_C = :DLR-DLR-DLR-C  
  END-EXEC
  + Tables Referenced: MEXW035\_DLR\_MSTR

1. **Select Retail Data from MEXW003/MEXW008 (5065P-SELECT-MEXW008-90V-DATA)**:

* EXEC SQL  
  SELECT A.VST\_STAT\_Y  
   ,B.VRS\_LCL\_FLEET\_C  
   ,B.VRS\_CST\_FIRST\_N  
   ,B.VRS\_CST\_BUS\_1\_N  
   ,B.VRS\_CST\_BUS\_2\_N  
   ,B.VRS\_CST\_MID\_INIT\_X  
   ,B.VRS\_CST\_LAST\_N  
   ,B.VRS\_CST\_ADDR\_1\_X  
   ,B.VRS\_CST\_ADD\_DIV2\_N  
   ,B.VRS\_CST\_ADD\_DIV1\_C  
   ,B.VRS\_CST\_POSTAL\_C  
   ,B.VRS\_SALESPERSON\_C  
   ,B.VRS\_TYP\_LCL\_CUST\_C  
   ,B.VRS\_RPT\_SALE\_Y  
   ,B.VRS\_WARR\_STRT\_Y  
  INTO :VST-STAT-Y  
   ,:VRS-LCL-FLEET-C  
   ,:VRS-CST-FIRST-N  
   ,:VRS-CST-BUS-1-N  
   ,:VRS-CST-BUS-2-N  
   ,:VRS-CST-MID-INIT-X  
   ,:VRS-CST-LAST-N  
   ,:VRS-CST-ADDR-1-X  
   ,:VRS-CST-ADD-DIV2-N  
   ,:VRS-CST-ADD-DIV1-C  
   ,:VRS-CST-POSTAL-C  
   ,:VRS-SALESPERSON-C  
   ,:VRS-TYP-LCL-CUST-C  
   ,:VRS-RPT-SALE-Y  
   ,:VRS-WARR-STRT-Y  
  FROM MEXW003\_VEH\_STATUS A  
   ,MEXW008\_VEH\_RTL B  
  WHERE A.VEH\_ORD\_ID\_C = :VST-VEH-ORD-ID-C  
   AND A.DTA\_DATA\_SRC\_C = :VST-DTA-DATA-SRC-C  
   AND A.STA\_STATUS\_C = :VST-STA-STATUS-C -- Input: PL-90V  
   AND A.VST\_LAST\_OCCUR\_F = :VST-LAST-OCCUR-F -- Input: PL-Y  
   AND A.VST\_ACTIVE\_F = :VST-ACTIVE-F -- Input: PL-Y  
   AND A.VST\_ACTIVE\_F = B.VRS\_ACTIVE\_F  
   AND A.STA\_STATUS\_C = B.STA\_STATUS\_C  
   AND A.VEH\_ORD\_ID\_C = B.VEH\_ORD\_ID\_C  
   AND A.DTA\_DATA\_SRC\_C = B.DTA\_DATA\_SRC\_C  
   AND SUBSTR(A.VST\_LCL\_LOC\_C, 1,7) = B.VRS\_LCL\_DLR\_C  
   AND A.VST\_STAT\_Y = B.VRS\_RETAIL\_Y  
  END-EXEC
  + Tables Referenced: MEXW003\_VEH\_STATUS, MEXW008\_VEH\_RTL

1. **Select Wholesale Data from MEXW003/MEXW007 (5085P-SELECT-MEXW003-40V)**:

* EXEC SQL  
  SELECT A.VST\_GBL\_LOC\_C  
   ,A.VST\_STAT\_Y  
   ,B.VWS\_TOT\_LCL\_A  
   ,B.CUR\_CURRENCY\_C  
  INTO :VST-GBL-LOC-C  
   ,:VST-STAT-Y  
   ,:VWS-TOT-LCL-A  
   ,:VWS-CUR-CURRENCY-C  
  FROM MEXW003\_VEH\_STATUS A  
   ,MEXW007\_VEH\_WHS B  
  WHERE A.VEH\_ORD\_ID\_C = :VST-VEH-ORD-ID-C  
   AND A.DTA\_DATA\_SRC\_C = :VST-DTA-DATA-SRC-C  
   AND A.STA\_STATUS\_C = :VST-STA-STATUS-C -- Input: PL-40V  
   AND A.VST\_LAST\_OCCUR\_F = :VST-LAST-OCCUR-F -- Input: PL-Y  
   AND A.VST\_ACTIVE\_F = :VST-ACTIVE-F -- Input: PL-Y  
   AND A.VST\_ACTIVE\_F = B.VWS\_ACTIVE\_F  
   AND A.STA\_STATUS\_C = B.STA\_STATUS\_C  
   AND A.VEH\_ORD\_ID\_C = B.VEH\_ORD\_ID\_C  
   AND A.DTA\_DATA\_SRC\_C = B.DTA\_DATA\_SRC\_C  
   AND A.VST\_STAT\_Y = B.VWS\_DATE\_Y  
   AND SUBSTR(A.VST\_LCL\_LOC\_C, 1,7) = B.VWS\_LCL\_DLR\_C  
  END-EXEC
  + Tables Referenced: MEXW003\_VEH\_STATUS, MEXW007\_VEH\_WHS

1. **Select from MEXW034\_VL\_BRAND (5110P-SELECT-W034-DATA)**:

* EXEC SQL  
  SELECT VLN\_WERS\_VL\_C  
   ,VLN\_WERS\_PRD\_TP\_C  
   ,VLN\_WERS\_BRAND\_C  
   ,VLN\_GEVIS\_VL\_N  
  INTO :VLN-WERS-VL-C  
   ,:VLN-WERS-PRD-TP-C  
   ,:VLN-WERS-BRAND-C  
   ,:VLN-GEVIS-VL-N  
  FROM MEXW034\_VL\_BRAND  
  WHERE DTA\_DATA\_SRC\_C = :VLN-DTA-DATA-SRC-C  
   AND VLN\_GEVIS\_VL\_C = :VLN-GEVIS-VL-C  
   AND VLN\_ACTIVE\_F = :VLN-ACTIVE-F -- Input: PL-Y  
  END-EXEC
  + Tables Referenced: MEXW034\_VL\_BRAND

1. **Select from MEXW032\_CATALOG (5112P-SELECT-MEXW032-WERS-VL)**:

* EXEC SQL  
  SELECT VHL\_VEH\_LINE\_C  
   ,VPT\_PROD\_TYP\_C  
  INTO :CTG-VHL-VEH-LINE-C  
   ,:CTG-VPT-PROD-TYP-C  
  FROM MEXW032\_CATALOG  
  WHERE DTA\_DATA\_SRC\_C = :CTG-DTA-DATA-SRC-C  
   AND CTG\_LCL\_CATALOG\_C = :CTG-LCL-CATALOG-C  
  END-EXEC
  + Tables Referenced: MEXW032\_CATALOG

1. **Select from MEXW034\_VL\_BRAND (5114P-SELECT-MEXW034-GEVIS-VL)**:

* EXEC SQL  
  SELECT VLN\_GEVIS\_VL\_C  
   ,VLN\_WERS\_PRD\_TP\_C  
   ,VLN\_WERS\_BRAND\_C  
   ,VLN\_GEVIS\_VL\_N  
  INTO :VLN-GEVIS-VL-C  
   ,:VLN-WERS-PRD-TP-C  
   ,:VLN-WERS-BRAND-C  
   ,:VLN-GEVIS-VL-N  
  FROM MEXW034\_VL\_BRAND  
  WHERE DTA\_DATA\_SRC\_C = :VLN-DTA-DATA-SRC-C  
   AND VLN\_WERS\_VL\_C = :VLN-WERS-VL-C  
   AND VLN\_ACTIVE\_F = :VLN-ACTIVE-F -- Input: PL-Y  
   AND VLN\_WERS\_PRD\_TP\_C = :VLN-WERS-PRD-TP-C  
  END-EXEC
  + Tables Referenced: MEXW034\_VL\_BRAND

1. **Select from MEXW033\_BODY\_TYPE (5115P-SELECT-MEXW033-DATA)**:

* EXEC SQL  
  SELECT BDT\_WERS\_BDY\_TYP\_C  
  INTO :BDT-WERS-BDY-TYP-C  
  FROM MEXW033\_BODY\_TYPE  
  WHERE BDT\_PROD\_SRC\_C = :BDT-PROD-SRC-C  
   AND BDT\_BDY\_TYP\_C = :BDT-BDY-TYP-C  
   AND BDT\_START\_YR\_R <= :BDT-START-YR-R  
   AND BDT\_END\_YR\_R >= :BDT-END-YR-R  
  END-EXEC
  + Tables Referenced: MEXW033\_BODY\_TYPE

1. **Select MEXW003 Status 20T (5120P-OBTAIN-MEXW003-20T)**:

* EXEC SQL  
  SELECT VST\_STAT\_Y  
  INTO :VST-STAT-Y  
  FROM MEXW003\_VEH\_STATUS  
  WHERE VEH\_ORD\_ID\_C = :VST-VEH-ORD-ID-C  
   AND DTA\_DATA\_SRC\_C = :VST-DTA-DATA-SRC-C  
   AND STA\_STATUS\_C = :VST-STA-STATUS-C -- Input: PL-20T  
   AND VST\_LAST\_OCCUR\_F = :VST-LAST-OCCUR-F -- Input: PL-Y  
   AND VST\_ACTIVE\_F = :VST-ACTIVE-F -- Input: PL-Y  
  END-EXEC
  + Tables Referenced: MEXW003\_VEH\_STATUS *(Similar SQL statements exist for statuses 30R, 30P, 30T, 80F in paragraphs 5140P, 5160P, 5180P, 5200P respectively, changing :VST-STA-STATUS-C value.)*

## 4.6 IMS Database Details

No IMS databases are referenced in the program.

## 4.7 Called Sub-routine/Program Details

* **EXWWSSTK**:
  + **Purpose**: Called in 2060P-CALL-EXWWSSTK to obtain the current stocking dealer (SSTK-GBL-STK-DLR-C), current stocking dealer status code (SSTK-CUR-STAT-C), and current stocking dealer status date (SSTK-CUR-STAT-Y) for a given vehicle order ID and data source.
  + **Interface**: Uses SSTK-I-O-DATA (defined in CPEWSSTK copybook).
    - Input: SSTK-MODE (set to ‘I’ for inquiry), SSTK-DTA-DATA-SRC-C, SSTK-VEH-ORD-ID-C.
    - Output: SSTK-GBL-STK-DLR-C, SSTK-LCL-STK-DLR-C, SSTK-STK-DLR-STAT-C, SSTK-STK-DLR-STAT-Y, SSTK-CUR-STAT-C, SSTK-CUR-STAT-Y, etc., and error/status fields in SSTK-OUT-DATA-MSG.
* **COREDUMP**:
  + **Purpose**: Called in 9999P-ABEND and 2070P-SSTK-FATAL-ERROR (if SSTK-SQL-RETURN-CODE is not +100) to force a program abend.
  + **Interface**: No parameters explicitly shown in the CALL statement within EXWWB911. Standard system abend.

## 4.8 VSAM File Details

No VSAM files are referenced in the program.

## 4.9 IBM MQ Details

No IBM MQ interfaces are referenced in the program.

## 4.10 CICS Details

This program is a batch program and does not involve CICS transactions.

## 4.11 Error Handling

* **Paragraph Name**: 0000P-MAINLINE
  + **Trigger Condition(s):**
    - SEND-EMAIL switch is ‘Y’ at the end of processing. This switch is set by 2070P-SSTK-FATAL-ERROR if EXWWSSTK returns SQLCODE +100, or by 9050P-MISSING-MEXW027-ROW if MEXW027\_CONV lookup fails.
  + **Action Taken:**
    - Sets RETURN-CODE to 3.
  + **Status Codes / Messages / Variables affected:**
    - RETURN-CODE = 3
* **Paragraph Name**: 2070P-SSTK-FATAL-ERROR (Called when EXWWSSTK returns SSTK-DB2-ERROR)
  + **Trigger Condition(s):**
    - EXWWSSTK subroutine call results in SSTK-DB2-ERROR.
  + **Action Taken:**
    - Writes detailed error information from SSTK-I-O-DATA (program ID, return code, paragraph, table names, host variables, SQL code, SQL messages) to AUDIT-FILE.
    - If SSTK-SQL-RETURN-CODE = +100:
      * Sets SEND-EMAIL switch to ‘Y’.
      * Increments PV-NBR-EXWWSSTK-NOTFOUND-CALLS.
    - Else (other SQL errors from EXWWSSTK):
      * Performs 9999C-CALL-COREDUMP to abend the program.
  + **Status Codes / Messages / Variables affected:**
    - AUDIT-FILE receives error details.
    - SEND-EMAIL = ‘Y’ (if +100).
    - PV-NBR-EXWWSSTK-NOTFOUND-CALLS incremented (if +100).
    - Program abend (if not +100).
* **Paragraph Name**: 2100P-OPEN-SALE-CHK-CSR
  + **Trigger Condition(s):**
    - SQLCODE is not OK after OPEN SALE\_CHK\_CSR.
  + **Action Taken:**
    - Populates DB2-ABEND-SQLCODE, DB2-ABEND-FUNCTION (“OPEN”), DB2-ABEND-TABLE (PL-SALE-CHK-CSR).
    - Moves DB2-ABEND-MSG to ABEND-MSG.
    - Sets ABEND-PARAGRAPH to “2100P”.
    - Performs 9999P-ABEND.
  + **Status Codes / Messages / Variables affected:**
    - Abend process initiated.
* **Paragraph Name**: 2120P-FETCH-SALE-CHK-CSR
  + **Trigger Condition(s):**
    - SQLCODE is not OK and not +100 (END-OF-CURSOR) after FETCH SALE\_CHK\_CSR.
  + **Action Taken:**
    - Populates DB2-ABEND-SQLCODE, DB2-ABEND-FUNCTION (“FETCH”), DB2-ABEND-TABLE (PL-SALE-CHK-CSR).
    - Moves DB2-ABEND-MSG to ABEND-MSG.
    - Sets ABEND-PARAGRAPH to “2120P”.
    - Performs 9999P-ABEND.
  + **Status Codes / Messages / Variables affected:**
    - Abend process initiated.
* **Paragraph Name**: 2140P-CLOSE-SALE-CHK-CSR
  + **Trigger Condition(s):**
    - SQLCODE is not OK after CLOSE SALE\_CHK\_CSR.
  + **Action Taken:**
    - Populates DB2-ABEND-SQLCODE, DB2-ABEND-FUNCTION (“CLOSE”), DB2-ABEND-TABLE (PL-SALE-CHK-CSR).
    - Moves DB2-ABEND-MSG to ABEND-MSG.
    - Sets ABEND-PARAGRAPH to “2140P”.
    - Performs 9999P-ABEND.
  + **Status Codes / Messages / Variables affected:**
    - Abend process initiated.
* **Paragraph Name**: 5045P-SELECT-MEXW027-DATA
  + **Trigger Condition(s):**
    - SQLCODE is not OK and not +100 (NOT-FOUND) after SELECT from MEXW027\_CONV.
  + **Action Taken:**
    - Populates DB2-ABEND-SQLCODE, DB2-ABEND-FUNCTION (“SELECT”), DB2-ABEND-TABLE (PL-TBL-CONV).
    - Moves DB2-ABEND-MSG to ABEND-MSG.
    - Sets ABEND-PARAGRAPH to “5045P”.
    - Performs 9999P-ABEND.
  + **Status Codes / Messages / Variables affected:**
    - Abend process initiated.
  + **Trigger Condition(s):** SQLCODE = +100 (NOT-FOUND)
  + **Action Taken:** Sets MEXW027-NOT-FOUND to TRUE. This is handled in 5040P-MOVE-SSTK-CURRENT-TO-DTL which then calls 9050P-MISSING-MEXW027-ROW.
* **Paragraph Name**: 5050P-GET-MEXW035-DATA
  + **Trigger Condition(s):**
    - SQLCODE is not OK and not +100 (NOT-FOUND) after SELECT from MEXW035\_DLR\_MSTR.
  + **Action Taken:**
    - Populates DB2-ABEND-SQLCODE, DB2-ABEND-FUNCTION (“SELECT”), DB2-ABEND-TABLE (PL-TBL-DLR-MSTR).
    - Moves DB2-ABEND-MSG to ABEND-MSG.
    - Sets ABEND-PARAGRAPH to “5050P”.
    - Performs 9999P-ABEND.
  + **Status Codes / Messages / Variables affected:**
    - Abend process initiated.
  + **Trigger Condition(s):** SQLCODE = +100 (NOT-FOUND)
  + **Action Taken:** Sets MEXW035-NOT-FOUND to TRUE. Performs 9000P-MISSING-MEXW035-ROW.
* **Paragraph Name**: 5065P-SELECT-MEXW008-90V-DATA (and similar DB2 SELECT paragraphs like 5085P, 5110P, 5112P, 5114P, 5115P, 5120P, 5140P, 5160P, 5180P, 5200P, 5220P (OPEN), 5230P (FETCH), 5240P (CLOSE), 5300P (OPEN), 5320P (FETCH), 5340P (CLOSE), 7000P, 7200P, 7250P (UPDATE), 7300P, 7350P (UPDATE), 7600P)
  + **Trigger Condition(s):**
    - DB2 SQLCODE is not OK and not +100 (NOT-FOUND or END-OF-CURSOR).
  + **Action Taken:**
    - Populates DB2-ABEND-SQLCODE, DB2-ABEND-FUNCTION (e.g., “SELECT”, “OPEN”, “FETCH”, “CLOSE”, “UPDATE”), DB2-ABEND-TABLE (specific table literal).
    - Moves DB2-ABEND-MSG to ABEND-MSG.
    - Sets ABEND-PARAGRAPH to the current paragraph name.
    - Performs 9999P-ABEND.
  + **Status Codes / Messages / Variables affected:**
    - Abend process initiated.
* **Paragraph Name**: 8000P-GET-SYSPARM-RECORD
  + **Trigger Condition(s):**
    - END-OF-SYSPARM-FILE is true and PV-NBR-SYSPARM-RECS-READ is zero (empty SYSPARM file).
  + **Action Taken:**
    - Moves “MISSING SYSPARM” to ABEND-MSG.
    - Moves “8000P” to ABEND-MSG-2.
    - Performs 9999P-ABEND.
  + **Status Codes / Messages / Variables affected:**
    - Abend process initiated.
* **Paragraph Name**: 9000P-MISSING-MEXW035-ROW
  + **Trigger Condition(s):**
    - Called when MEXW035\_DLR\_MSTR SELECT results in SQLCODE +100.
  + **Action Taken:**
    - Writes “MISSING DEALER ON MEXW035:”, “DEALER CODE: ”, “ASSOCIATED VIN: ” to AUDIT-FILE.
    - Increments PV-NBR-MEXW035-NOTFOUND-CALLS.
  + **Status Codes / Messages / Variables affected:**
    - AUDIT-FILE updated. PV-NBR-MEXW035-NOTFOUND-CALLS incremented.
* **Paragraph Name**: 9050P-MISSING-MEXW027-ROW
  + **Trigger Condition(s):**
    - Called when MEXW027\_CONV SELECT results in SQLCODE +100 (via 5040P).
  + **Action Taken:**
    - Writes “MISSING STATUS ON MEXW027:”, “STATUS CODE: ”, “ASSOCIATED VIN: ” to AUDIT-FILE.
    - Sets SEND-EMAIL switch to ‘Y’.
    - Increments PV-NBR-MEXW027-NOTFOUND-CALLS.
  + **Status Codes / Messages / Variables affected:**
    - AUDIT-FILE updated. SEND-EMAIL = ‘Y’. PV-NBR-MEXW027-NOTFOUND-CALLS incremented.
* **Paragraph Name**: 9999P-ABEND
  + **Trigger Condition(s):**
    - Called by various paragraphs upon detecting unrecoverable errors (e.g., DB2 errors, missing critical input).
  + **Action Taken:**
    - Ensures ABEND-PROGRAM is set (defaults to PL-PGM-NAME).
    - Writes ABEND-MSG and ABEND-MSG-2 (containing program and paragraph info) to AUDIT-FILE.
    - Performs 9999C-CALL-COREDUMP to force an abend.
  + **Status Codes / Messages / Variables affected:**
    - AUDIT-FILE updated with abend messages. Program abends.

# 5. Interface Design

## 5.1 External Interfaces

* **Input SYSPARM File (PARMFILE)**: A sequential file providing a list of producer codes (data sources) for processing. Each record specifies a DTA\_DATA\_SRC\_C.
* **Output Marketing Associates File (MKTASSOC)**:
  + Dataset Name: FDEX.WW.PROD.EXWWB490.MKTASC.FLTSLS
  + Format: A sequential file with a 1000-byte record length. Contains a HUB header record, multiple detail records based on WS-MKT-ASSOC-DETAIL-RECORD, and a HUB trailer record.
  + Transfer: Picked up by Marketing Associates via FTP using FTP ID FDFTPMK.
* **Audit File (AUDIT)**: A sequential file used for logging processing statistics, error messages, and abend information.
* **DB2 Database**: The program extensively reads from and updates (control table MEXS016\_GENERIC2) several DB2 tables. Key tables include MEXW001\_VEH\_ORDER, MEXW008\_VEH\_RTL, MEXW003\_VEH\_STATUS, MEXW035\_DLR\_MSTR, MEXW027\_CONV, and others as detailed in section 4.5.
* **Sub-programs**:
  + EXWWSSTK: Called to retrieve current stocking dealer information.
  + COREDUMP: Called to force a program abend in case of severe errors.

## 5.2 User Interface

This is a batch program and has no direct user interface.

# 6. Testing Strategy

## 6.1 Test Plan

* **Unit Testing**: Test individual paragraphs, especially those with complex logic or DB2 interactions (e.g., 2050P-VERIFY-WDMO-DEALER, 5000P-PROCESS-GEVIS-DETAIL-REC, individual DB2 access paragraphs). Mock EXWWSSTK calls. Use varied SYSPARM inputs and DB2 test data to cover different scenarios, including empty tables, data not found, and various status code combinations.
* **Integration Testing**: Test the interaction between EXWWB911 and the EXWWSSTK subroutine. Test DB2 interactions with a populated test database reflecting production-like data. Verify MEXS016\_GENERIC2 update logic.
* **System Testing**: Process sample SYSPARM files and verify the correctness of the MKT-ASSOC-FILE content and format, including HUB headers/trailers. Validate audit file content and error handling, including the generation of CC3 and email trigger for specific error conditions. Ensure FTP process can pick up the file.
* **Regression Testing**: Use existing test cases (if available) and create new ones to ensure existing functionality is not broken by changes.

## 6.2 Testing Environment

Testing will be conducted in a mainframe test environment with: \* Access to a test DB2 subsystem with copies of required tables (MEXW001, MEXW008, MEXW003, etc.). \* Test versions of input SYSPARM-FILE. \* The EXWWSSTK and COREDUMP subroutines available in the test load libraries. \* JCL to execute the EXWWB911 batch program. \* Tools to view and validate output files (MKT-ASSOC-FILE, AUDIT-FILE) and DB2 table contents.

# 7. Appendices

## 7.1 Glossary

* **GEVIS**: Acronym for a system providing vehicle data (likely Global EXport Vehicle Information System or similar).
* **WDMO**: Wholesale Dealer Marketing Organization. A classification for dealers.
* **SYSPARM**: System Parameter file, used here to input producer codes.
* **MKT-ASSOC-FILE**: The primary output file for Marketing Associates.
* **SALE\_CHK\_CSR**: Main DB2 cursor driving the extraction logic.
* **EXWWSSTK**: Subroutine to get current stocking dealer information.
* **WERS**: WERS (Worldwide Engineering Release System) string or related data, vehicle configuration/feature information.
* **FTP**: File Transfer Protocol, used by Marketing Associates to retrieve the output file.
* **HUB Header/Trailer**: Standardized header and trailer records for data exchange files.
* **DCLGEN**: DB2 Declaration Generator, used to create COBOL copybooks for DB2 table layouts.

## 7.2 References

* COBOL Program: EXWWB911
* Copybooks:
  + CPEWD001 (MEXW001\_VEH\_ORDER DCLGEN)
  + CPEWD003 (MEXW003\_VEH\_STATUS DCLGEN)
  + CPEWD004 (MEXW004\_VEH\_WERS\_STRING DCLGEN)
  + CPEWD007 (MEXW007\_VEH\_WHS DCLGEN)
  + CPEWD008 (MEXW008\_VEH\_RTL DCLGEN)
  + CPESD016 (MEXS016\_GENERIC2 DCLGEN)
  + CPEWD021 (MEXW021\_SUBLVL\_ASG DCLGEN)
  + CPEWD027 (MEXW027\_CONV DCLGEN)
  + CPEWD031 (MEXW031\_CATMAP DCLGEN)
  + CPEWD032 (MEXW032\_CATALOG DCLGEN)
  + CPEWD033 (MEXW033\_BODY\_TYPE DCLGEN)
  + CPEWD034 (MEXW034\_VL\_BRAND DCLGEN)
  + CPEWD035 (MEXW035\_DLR\_MSTR DCLGEN)
  + CPESDB2 (SQLCA and SQLCODES)
  + CPESGNTB (Generic Table Layouts - EXSE System)
  + CPEWGNTB (Generic Table Layout - EXWW System)
  + CPESEBWS (BMPSHELL Working Storage, Audit Formatting)
  + CPEWSSTK (I/O Parameters for Subroutine EXWWSSTK)
  + CPEWHUB (E & G HUB Header/Trailer Layout)
* Called Subroutines: EXWWSSTK, COREDUMP

|  |
| --- |
| End of COBOL Technical Design Specification for Modernization |