THIS IS THE MAIN CODE

```
USE [Wild West];

-- Step 1: Create a new table 'Month_Data_8' and insert January data into it
SELECT *
INTO dbo.Month_Data_8  -- Create the table 'Month_Data_8' and insert data for January 2024
FROM Portfolio_clean
WHERE MONTH(Sales_Date) = 1;

-- Step 2: Insert data for February to August into the existing 'Month_Data_8' table
INSERT INTO dbo.Month_Data_8
SELECT *
FROM Portfolio_clean
WHERE MONTH(Sales_Date) IN (2, 3, 4, 5, 6, 7, 8);

-- Step 3: Ensure that the target table has the correct structure
-- Add 'Commission' column if it doesn't exist
IF NOT EXISTS (SELECT * FROM INFORMATION_SCHEMA.COLUMNS
        WHERE TABLE_NAME = 'Month_Data_8' AND COLUMN_NAME = 'Commission')
BEGIN
  ALTER TABLE dbo.Month_Data_8
  ADD Commission DECIMAL(18, 2);
END;

-- Add 'Commission_Eligibility' column if it doesn't exist
IF NOT EXISTS (SELECT * FROM INFORMATION_SCHEMA.COLUMNS
        WHERE TABLE_NAME = 'Month_Data_8' AND COLUMN_NAME =
'Commission_Eligibility')
BEGIN
```

```
  ALTER TABLE dbo.Month_Data_8
  ADD Commission_Eligibility VARCHAR(3);
END;

-- Add 'Agm_Sts_Cd' column if it doesn't exist
IF NOT EXISTS (SELECT * FROM INFORMATION_SCHEMA.COLUMNS
        WHERE TABLE_NAME = 'Month_Data_8' AND COLUMN_NAME =
'Agm_Sts_Cd')
BEGIN
  ALTER TABLE dbo.Month_Data_8
  ADD Agm_Sts_Cd VARCHAR(50);
END;

-- Add 'Org_Lvl_Nm' column if it doesn't exist
IF NOT EXISTS (SELECT * FROM INFORMATION_SCHEMA.COLUMNS
        WHERE TABLE_NAME = 'Month_Data_8' AND COLUMN_NAME =
'Org_Lvl_Nm')
BEGIN
  ALTER TABLE dbo.Month_Data_8
  ADD Org_Lvl_Nm VARCHAR(50);
END;

-- Add 'Fld_Rep_Cd' column if it doesn't exist
IF NOT EXISTS (SELECT * FROM INFORMATION_SCHEMA.COLUMNS
        WHERE TABLE_NAME = 'Month_Data_8' AND COLUMN_NAME = 'Fld_Rep_Cd')
BEGIN
  ALTER TABLE dbo.Month_Data_8
  ADD Fld_Rep_Cd VARCHAR(50);
END;

-- Add 'pev_portfolioResponsibleCode' column if it doesn't exist
IF NOT EXISTS (SELECT * FROM INFORMATION_SCHEMA.COLUMNS
```

```
        WHERE TABLE_NAME = 'Month_Data_8' AND COLUMN_NAME =
'pev_portfolioResponsibleCode')
BEGIN
  ALTER TABLE dbo.Month_Data_8
  ADD pev_portfolioResponsibleCode VARCHAR(50);
END;

-- Add 'pev_id' column if it doesn't exist
IF NOT EXISTS (SELECT * FROM INFORMATION_SCHEMA.COLUMNS
        WHERE TABLE_NAME = 'Month_Data_8' AND COLUMN_NAME = 'pev_id')
BEGIN
  ALTER TABLE dbo.Month_Data_8
  ADD pev_id VARCHAR(50);
END;

-- Add 'policy_type' column if it doesn't exist
IF NOT EXISTS (SELECT * FROM INFORMATION_SCHEMA.COLUMNS
        WHERE TABLE_NAME = 'Month_Data_8' AND COLUMN_NAME = 'policy_type')
BEGIN
  ALTER TABLE dbo.Month_Data_8
  ADD policy_type VARCHAR(50);
END;

-- Add 'agent_performance' column if it doesn't exist
IF NOT EXISTS (SELECT * FROM INFORMATION_SCHEMA.COLUMNS
        WHERE TABLE_NAME = 'Month_Data_8' AND COLUMN_NAME =
'agent_performance')
BEGIN
  ALTER TABLE dbo.Month_Data_8
  ADD agent_performance VARCHAR(50);
END;
```

```
-- Step 4: Add 'Policy_Status' column if it doesn't already exist
IF COL_LENGTH('dbo.Month_Data_8', 'Policy_Status') IS NULL
BEGIN
  ALTER TABLE dbo.Month_Data_8
  ADD Policy_Status VARCHAR(10);
END;

-- Step 5: Add necessary columns to 'Month_Data_8' if they don't already exist
-- Add 'Season' column to store the season (Q1, Q2, Q3, Q4)
IF COL_LENGTH('dbo.Month_Data_8', 'Season') IS NULL
BEGIN
  ALTER TABLE dbo.Month_Data_8
  ADD Season VARCHAR(5);
END;

-- Add 'Seasonal_Performance' column to store performance indicator
IF COL_LENGTH('dbo.Month_Data_8', 'Seasonal_Performance') IS NULL
BEGIN
  ALTER TABLE dbo.Month_Data_8
  ADD Seasonal_Performance VARCHAR(20);
END;

-- Ensure 'Clawback_Category' column exists in 'Month_Data_8'
IF COL_LENGTH('dbo.Month_Data_8', 'Clawback_Category') IS NULL
BEGIN
  ALTER TABLE dbo.Month_Data_8
  ADD Clawback_Category VARCHAR(20); -- To store clawback timing categories like 'Early', 'Mid-Term', 'Late'
END;

-- Step 6: Update 'Policy_Status' based on 'Policy_Duration' in 'Month_Data_8'
UPDATE dbo.Month_Data_8
```

```
SET Policy_Status = CASE
  WHEN Policy_Duration >= 365 THEN 'Active'
  ELSE 'Cancel'
END;

-- Step 7: Create a temporary table to gather policy and transaction details
WITH PolicyTransactions AS (
  SELECT
    p.Key_Policy,
    p.bnd_dt,
    p.pln_end_dt,
    p.Vld_Fm_Tms,
    p.Vld_To_Tms,
    p.Agrm_Sts_Cd,
    pt.pev_createdat,
    pt.pev_portfolioresponsiblecode,
    pt.pev_id,
    p.Vrsn,
    p.Sub_Vrsn,
    DATEDIFF(day, p.Vld_Fm_Tms , p.Vld_To_Tms) AS Policy_Duration,
    DATEADD(day, -1, p.Sts_Dt) AS Cncl_Dt
  FROM dbo.Policy_clean p
  JOIN dbo.Policy_Transactions_clean pt ON p.Ext_Refr = pt.po_No
  WHERE
    p.Agrm_Sts_Cd IN ('Active', 'Cancel')
    AND p.Vld_Fm_Tms BETWEEN '2024-01-01' AND '2024-08-31'
    AND p.Vld_To_Tms IS NOT NULL
    AND p.Vrsn = (SELECT MAX(Vrsn) FROM dbo.Policy_clean WHERE Key_Policy =
p.Key_Policy)
    AND p.Sub_Vrsn = (SELECT MAX(Sub_Vrsn) FROM dbo.Policy_clean WHERE
Key_Policy = p.Key_Policy AND Vrsn = p.Vrsn)
    AND pt.pev_createdat BETWEEN p.Vld_Fm_Tms AND p.Vld_To_Tms
```

```
),

-- Step 8: Filter consistent portfolio codes for 'WILDWEST-2' and 'WILDWEST-3'
ConsistentPortfolioCodes AS (
  SELECT Key_Policy, MIN(pev_portfolioresponsiblecode) AS PortfolioCode
  FROM PolicyTransactions
  GROUP BY Key_Policy
  HAVING COUNT(DISTINCT pev_portfolioresponsiblecode) = 1
    AND MIN(pev_portfolioresponsiblecode) IN ('WILDWEST-2', 'WILDWEST-3')
),

-- Step 9: Prepare 'SalesPortfolio' with relevant policy data
SalesPortfolio AS (
  SELECT
    po.Key_Policy,
    po.Payment_Status,
    po.Product_Name,
    po.Sales_Date,
    po.Vld_Fm_Tms,
    po.Vld_To_Tms,
    DATEDIFF(day, po.Vld_Fm_Tms, po.Vld_To_Tms) AS Policy_Duration,
    s.Org_Lvl_Nm,
    pt.pev_portfolioresponsiblecode,
    po.Annual_Premium,
    pt.Agrm_Sts_Cd,
    pt.Cncl_Dt,
    ROW_NUMBER() OVER (PARTITION BY po.Key_Policy ORDER BY po.Sales_Date
DESC) AS RowNum,
    CASE
      WHEN po.Annual_Premium >= 1500 THEN 'Premium'
      WHEN DATEDIFF(day, po.Vld_Fm_Tms, po.Vld_To_Tms) > 365 THEN 'Long-Term'
      ELSE 'Standard'
```

```
        END AS Policy_Type,
        CASE
            WHEN po.Annual_Premium >= 1000 THEN 'Good'
            ELSE 'Bad'
        END AS Agent_Performance
    FROM dbo.Portfolio_clean po
    JOIN dbo.Sales_Org_clean s ON po.Key_SS_Org = s.Key_SS_Org
    JOIN PolicyTransactions pt ON po.Key_Policy = pt.Key_Policy
    JOIN ConsistentPortfolioCodes cpc ON pt.Key_Policy = cpc.Key_Policy
    WHERE
        po.Payment_Status = 'Paid'
        AND (
            (s.Org_Lvl_Nm = 'Outbound' AND pt.pev_portfolioresponsiblecode = 'WILDWEST-3')
            OR (s.Org_Lvl_Nm = 'Internet' AND pt.pev_portfolioresponsiblecode = 'WILDWEST-3')
            OR (s.Org_Lvl_Nm = 'Inbound' AND pt.pev_portfolioresponsiblecode = 'WILDWEST-2')
        )
        AND s.fld_rep_cd NOT LIKE 'Inactive%'
),

-- Step 10: Determine the season for cancellation data
CancellationData AS (
    SELECT
        sp.Key_Policy,
        CASE
            WHEN MONTH(sp.Sales_Date) IN (12, 1, 2) THEN 'Q1'
            WHEN MONTH(sp.Sales_Date) IN (3, 4, 5) THEN 'Q2'
            WHEN MONTH(sp.Sales_Date) IN (6, 7, 8) THEN 'Q3'
            WHEN MONTH(sp.Sales_Date) IN (9, 10, 11) THEN 'Q4'
        END AS Season
    FROM SalesPortfolio sp
    GROUP BY sp.Key_Policy, sp.Sales_Date
```

```
)

-- Step 11: Final selection with calculated values (e.g., Commission, Season, Policy_Status)
SELECT
    sp.Key_Policy,
    sp.Org_Lvl_Nm,
    sp.Annual_Premium,
    sp.Product_Name,
    sp.pev_portfolioresponsiblecode,
    sp.Policy_Duration,
    sp.Payment_Status,
    sp.Sales_Date,
    -- Updated Commission_Amount calculation
    CASE
        WHEN sp.Sales_Date BETWEEN '2024-01-01' AND '2024-01-31' THEN 0 -- Commission is 0 for January 2024
        WHEN ROW_NUMBER() OVER (ORDER BY sp.Key_Policy) <= 1500 THEN 0.12 * sp.Annual_Premium
        ELSE 0.14 * sp.Annual_Premium
    END AS Commission_Amount,
    cd.Season
FROM SalesPortfolio sp
LEFT JOIN CancellationData cd ON sp.Key_Policy = cd.Key_Policy
WHERE sp.RowNum = 1;

-- Step 12: Update the 'EMD' table with calculated values such as Policy_Status, Commission_Eligibility, and Clawback
UPDATE EMD
SET
    Policy_Status = CASE WHEN sp.Policy_Duration >= 365 THEN 'Active' ELSE 'Cancel' END,
    Policy_Duration = COALESCE(sp.Policy_Duration, 0),
    Commission_Eligibility = CASE
```

```
    WHEN sp.Sales_Date BETWEEN '2024-01-01' AND '2024-01-31' THEN 'NO'
    ELSE
      CASE
        WHEN sp.pev_portfolioresponsiblecode IN ('WILDWEST-2', 'WILDWEST-3')
          AND (sp.Vld_Fm_Tms <= EMD.Sales_Date AND (sp.Vld_To_Tms IS NULL OR
sp.Vld_To_Tms >= EMD.Sales_Date))
          AND sp.Payment_Status = 'Paid'
          AND (sp.Product_Name BETWEEN 'Product 1' AND 'Product 8' OR
sp.Product_Name BETWEEN 'Product 13' AND 'Product 31')
          AND (
            (sp.pev_portfolioresponsiblecode = 'WILDWEST-3' AND (sp.Org_Lvl_Nm =
'Outbound' OR sp.Org_Lvl_Nm = 'Internet'))
            OR (sp.pev_portfolioresponsiblecode = 'WILDWEST-2' AND sp.Org_Lvl_Nm =
'Inbound')
          )
        THEN 'YES'
        ELSE 'NO'
      END,
  Cncl_Dt = CASE WHEN Policy_Status = 'Cancel' THEN sp.Cncl_Dt ELSE NULL END,
  -- Updated Clawback calculation
  Clawback = CASE
    WHEN sp.Sales_Date BETWEEN '2024-01-01' AND '2024-01-31' THEN 0 -- Clawback is
0 for January 2024
    WHEN Policy_Status = 'Cancel'
      AND sp.Cncl_Dt IS NOT NULL
      AND DATEDIFF(day, sp.Vld_Fm_Tms, sp.Vld_To_Tms) > 0
      AND (EMD.Sales_Date < '2024-01-01' OR EMD.Sales_Date > '2024-01-31')
    THEN
      ABS(
        (CAST(DATEDIFF(day, sp.Vld_Fm_Tms, sp.Cncl_Dt) AS FLOAT) /
         CAST(DATEDIFF(day, sp.Vld_Fm_Tms, sp.Vld_To_Tms) AS FLOAT)) *
sp.Annual_Premium
      )
```

```
        ELSE 0
    END,
    EMD.Season = cd.Season,
    EMD.Seasonal_Performance = CASE
        WHEN cd.Cancellation_Rate < 10 THEN 'Good'
        WHEN cd.Cancellation_Rate BETWEEN 10 AND 20 THEN 'Average'
        ELSE 'Poor'
    END
FROM SalesPortfolio sp
LEFT JOIN CancellationData cd ON sp.Key_Policy = cd.Key_Policy
WHERE sp.RowNum = 1;
```

CODE FOR AGGREGATE MONTHLY ANALYSIS :

```
-- Create a new table to store the summarized data with Commission Eligibility for the
Wild West dataset
CREATE TABLE [Wild West].dbo.Policy_Summary_Updated (
    Month VARCHAR(7),  -- Month-Year format
    Total_Policies INT,
    Active_Policies INT,
    Cancelled_Policies INT,
    New_Policies INT,
    Renewed_Policies INT,
    Premium_Revenue DECIMAL(18,2),
    Commission_Paid DECIMAL(18,2),
    Total_Clawback DECIMAL(18,2),
    Premium_Policies INT,
    Standard_Policies INT,
    Long_Term_Policies INT,
    Outbound_Sales INT,
    Internet_Sales INT,
    Inbound_Sales INT,
    Good_Performance_Agents INT,
    Poor_Performance_Agents INT,
    Seasonal_Performance VARCHAR(10),
    Commission_Eligibility VARCHAR(10)  -- New column for Commission Eligibility
);
SELECT
    FORMAT(Sales_Date, 'yyyy-MM') AS Month,  -- Month-Year format
    COUNT(Key_Policy) AS Total_Policies,
    SUM(CASE WHEN Policy_Status = 'Active' THEN 1 ELSE 0 END) AS Active_Policies,
    SUM(CASE WHEN Policy_Status = 'Cancel' THEN 1 ELSE 0 END) AS Cancelled_Policies,
    SUM(CASE WHEN Policy_Type = 'New' THEN 1 ELSE 0 END) AS New_Policies,
    SUM(CASE WHEN Policy_Type = 'Renewed' THEN 1 ELSE 0 END) AS Renewed_Policies,
    SUM(Annual_Premium) AS Premium_Revenue,
    SUM(Commission) AS Commission_Paid,
    SUM(Clawback) AS Total_Clawback,  -- Total Clawback
    SUM(CASE WHEN Policy_Type = 'Premium' THEN 1 ELSE 0 END) AS Premium_Policies,
    SUM(CASE WHEN Policy_Type = 'Standard' THEN 1 ELSE 0 END) AS Standard_Policies,
    SUM(CASE WHEN Policy_Type = 'Long-Term' THEN 1 ELSE 0 END) AS Long_Term_Policies,
    SUM(CASE WHEN Org_Lvl_Nm = 'Outbound' THEN 1 ELSE 0 END) AS Outbound_Sales,
    SUM(CASE WHEN Org_Lvl_Nm = 'Internet' THEN 1 ELSE 0 END) AS Internet_Sales,
    SUM(CASE WHEN Org_Lvl_Nm = 'Inbound' THEN 1 ELSE 0 END) AS Inbound_Sales,
    SUM(CASE WHEN Agent_Performance = 'Good' THEN 1 ELSE 0 END) AS
Good_Performance_Agents,
    SUM(CASE WHEN Agent_Performance = 'Bad' THEN 1 ELSE 0 END) AS
Poor_Performance_Agents,
    CASE
        WHEN AVG(Annual_Premium) >= 1500 THEN 'Good'
        WHEN AVG(Annual_Premium) BETWEEN 1000 AND 1499 THEN 'Average'
        ELSE 'Poor'
    END AS Seasonal_Performance
FROM dbo.Month_Data_8
WHERE Sales_Date BETWEEN '2024-01-01' AND '2024-08-31'
  AND Commission_Eligibility = 'YES'  -- Filter only rows with Commission Eligibility
= 'YES'
GROUP BY FORMAT(Sales_Date, 'yyyy-MM')
ORDER BY Month ASC;
```

CODE FOR EXTRA ANALYSIS : MONTHLY BREAKDOWN OF ACTIVE VS CANCELED POLICIES

```
USE [Wild West]
-- Create a new table and insert the result of the query
SELECT
    EMD.Sales_Date,                             -- The specific date for the policy
    EMD.Seasonal_Performance,                   -- Seasonal performance (Good, Average,
Poor)
    EMD.Policy_Status,                          -- Policy status (Active/Cancel) on the
given date
    COUNT(*) AS PolicyCount,                    -- Count of policies for the specific
date
    SUM(CASE WHEN EMD.Policy_Status = 'Active' THEN 1 ELSE 0 END) AS ActivePolicies,
-- Count of Active policies
    SUM(CASE WHEN EMD.Policy_Status = 'Cancel' THEN 1 ELSE 0 END) AS CanceledPolicies,
-- Count of Canceled policies
    AVG(EMD.Cancellation_Rate) AS AvgCancellationRate   -- Average cancellation rate
for that specific date
INTO dbo.PolicySeasonalSummary   -- This creates the new table
FROM dbo.Month_Data_8 EMD
GROUP BY
    EMD.Sales_Date,                             -- Group by specific date
    EMD.Seasonal_Performance,                   -- Group by Seasonal Performance on
that specific date
    EMD.Policy_Status                            -- Group by Policy Status
(Active/Cancel) on that specific date
ORDER BY
    EMD.Sales_Date,                             -- Order results by Sales Date
    EMD.Seasonal_Performance,                   -- Then by Seasonal Performance (Good,
Average, Poor)
    EMD.Policy_Status;                          -- Finally, by Policy Statustus
(Active/Cancel)
```

CODE FOR THE DATA MODEL

```
use [Wild West]
CREATE TABLE Policy (
	Key_Policy INT PRIMARY KEY,
	Ext_Refr NVARCHAR(50),
	Term TINYINT,
	Vrsn TINYINT,
	Sub_Vrsn TINYINT,
	Agrm_Sts_Cd NVARCHAR(10),
	Prtn NVARCHAR(50)
);
CREATE TABLE PolicyDates (
	PolicyDate_ID INT PRIMARY KEY IDENTITY,
	Key_Policy INT REFERENCES Policy(Key_Policy),
	Incp_Dt DATE,
	Bnd_Dt DATE,
	Pln_End_Dt DATE,
	Rnew_Dt DATE,
	Cncl_Dt DATE,
	Sts_Dt DATE
);
CREATE TABLE SalesOrganization (
	Key_SS_Org INT,
	Key_Policy INT REFERENCES Policy(Key_Policy),
	Fld_Rep_Cd NVARCHAR(10),
	Org_Lvl_cd NVARCHAR(10),
	Org_Lvl_Nm NVARCHAR(50)
);
CREATE TABLE Portfolio (
	Portfolio_ID INT PRIMARY KEY IDENTITY,
	Key_Policy INT REFERENCES Policy(Key_Policy),
	Key_SS_Org INT,
	Sales_Date DATE,
	Annual_Premium BIGINT
);
CREATE TABLE PortfolioValidity (
	Validity_ID INT PRIMARY KEY IDENTITY,
	Portfolio_ID INT REFERENCES Portfolio(Portfolio_ID),
	Vld_Fm_Tms DATETIME2,
	Vld_To_Tms DATETIME2
);
CREATE TABLE PortfolioStatus (
	PortfolioStatus_ID INT PRIMARY KEY IDENTITY,
	Portfolio_ID INT REFERENCES Portfolio(Portfolio_ID),
	Payment_Status NVARCHAR(20),
	PaymentDate DATE
);
CREATE TABLE Product (
	Product_ID INT PRIMARY KEY IDENTITY,
	product_Name NVARCHAR(50)
);
CREATE TABLE PortfolioProduct (
	PortfolioProduct_ID INT PRIMARY KEY IDENTITY,
	Portfolio_ID INT REFERENCES Portfolio(Portfolio_ID),
	Product_ID INT REFERENCES Product(Product_ID),
	No_Of_Sold_Policies TINYINT,
	No_Of_Cancelled_Policies INT,
	No_Of_Paid_Policies INT
```

```
	);
	CREATE TABLE [Transaction] (
		Transaction_ID INT PRIMARY KEY IDENTITY,
		Created_At DATETIME2,
		Portfolio_Responsible_Code NVARCHAR(50),
		Key_Policy INT REFERENCES Policy(Key_Policy),
		Portfolio_No NVARCHAR(50)
	);
	CREATE TABLE CancellationPolicy (
		Key_Policy INT REFERENCES Policy(Key_Policy),
		Cancel_Before_Vld_Frm INT,
		Cancel_After_Vld_Frm INT
	);
	CREATE TABLE TransactionResponsible (
		Responsible_ID INT PRIMARY KEY IDENTITY,
		Transaction_ID INT REFERENCES [Transaction](Transaction_ID),
		Pev_PortfolioResponsibleCode NVARCHAR(50)
	);
	use [Wild West]
	-- Insert statements with updated column names
	INSERT INTO Policy (Key_Policy, Ext_Refr, Term, Vrsn, Sub_Vrsn, Agrm_Sts_Cd,
Prtn)
	SELECT Key_Policy, Ext_Refr, Term, Vrsn, Sub_Vrsn, Agrm_Sts_Cd, Prtn FROM
dbo.Policy_clean;
	INSERT INTO PolicyDates (Key_Policy, Incp_Dt, Bnd_Dt, Pln_End_Dt, Rnew_Dt,
Cncl_Dt, Sts_Dt)
	SELECT Key_Policy, Incp_Dt, Bnd_Dt, Pln_End_Dt, Rnew_Dt, Cncl_Dt, Sts_Dt FROM
dbo.Policy_clean;
	INSERT INTO SalesOrganization (Key_SS_Org, Key_Policy, Fld_Rep_Cd, Org_Lvl_cd,
Org_Lvl_Nm)
	SELECT DISTINCT Sales_Org_clean.Key_SS_Org, dbo.Policy_clean.Key_Policy,
Sales_Org_clean.Fld_Rep_Cd, Sales_Org_clean.Org_Lvl_cd, Sales_Org_clean.Org_Lvl_Nm
	FROM Sales_Org_clean
	INNER JOIN Portfolio_clean ON Sales_Org_clean.Key_SS_Org =
Portfolio_clean.Key_SS_Org
	INNER JOIN Policy_clean ON Portfolio_clean.Key_Policy =
Policy_clean.Key_Policy;
	INSERT INTO Portfolio (Key_Policy, Key_SS_Org, Sales_Date, Annual_Premium)
	SELECT lp.Key_Policy, lp.Key_SS_Org, lp.Sales_Date, lp.Annual_Premium FROM
dbo.Portfolio_clean lp;
	INSERT INTO PortfolioValidity (Portfolio_ID, Vld_Fm_Tms, Vld_To_Tms)
	SELECT p.Portfolio_ID, lp.Vld_Fm_Tms, lp.Vld_To_Tms
	FROM dbo.Portfolio_clean lp
	INNER JOIN Portfolio p ON lp.Key_Policy = p.Key_Policy;
	INSERT INTO PortfolioStatus (Portfolio_ID, Payment_Status, PaymentDate)
	SELECT p.Portfolio_ID, lp.Payment_Status, lp.PaymentDate
	FROM dbo.Portfolio_clean lp
	INNER JOIN Portfolio p ON lp.Key_Policy = p.Key_Policy;
	INSERT INTO Product (product_Name)
	SELECT DISTINCT product_Name FROM dbo.Portfolio_clean WHERE product_Name IS NOT
NULL;
	INSERT INTO PortfolioProduct (Portfolio_ID, Product_ID, No_Of_Sold_Policies,
No_Of_Cancelled_Policies, No_Of_Paid_Policies)
	SELECT p.Portfolio_ID, pr.Product_ID, lp.No_Of_Sold_Policies,
lp.No_Of_Cancelled_Policies, lp.No_Of_Paid_Policies
	FROM dbo.Portfolio_clean lp
	INNER JOIN Portfolio p ON lp.Key_Policy = p.Key_Policy
	INNER JOIN Product pr ON lp.product_Name = pr.product_Name;
	INSERT INTO [Transaction] (Created_At, Portfolio_Responsible_Code, Key_Policy,
Portfolio_No)
	SELECT pev_CreatedAt, Pev_PortfolioResponsibleCode, Key_Policy, po_No FROM
dbo.Policy_Transactions_clean;
```

```
	INSERT INTO CancellationPolicy (Key_Policy, Cancel_Before_Vld_Frm,
Cancel_After_Vld_Frm)
	SELECT Key_Policy, Cancel_Before_Vld_Frm, Cancel_After_Vld_Frm FROM
dbo.Portfolio_clean;
	INSERT INTO TransactionResponsible (Transaction_ID,
Pev_PortfolioResponsibleCode)
	SELECT t.Transaction_ID, lt.Pev_PortfolioResponsibleCode
	FROM Policy_Transactions_clean AS lt
	INNER JOIN [Transaction] AS t ON lt.Key_Policy = t.Key_Policy;
```