These are the analytical views you defined for performance, KPIs, and discount analysis. I selected 14 representative ones that cover store, customer, product, employee, discount, and business KPIs.

```
-- Store Performance Views
CREATE VIEW vwRevenueByStore AS
SELECT
   s.[Store_ID], s.[Store_Name], s.[Country], s.[City],
   SUM(t.[Line_Total]) AS StoreRevenue,
   COUNT(DISTINCT t.[Invoice_ID]) AS TotalInvoices,
   AVG(t.[Invoice_Total]) AS AvgInvoiceValue
FROM [Transactions] t
JOIN Stores s ON t.[Store_ID] = s.[Store_ID]
GROUP BY s.[Store_ID], s.[Store_Name], s.[Country], s.[City];

-- Customer Performance Views
CREATE VIEW vwCustomerOverview AS
SELECT
   c.Customer_ID, c.Name,
   c.Country,
   COUNT(t.Invoice_ID) AS TotalOrders,
   SUM(t.Invoice_Total) AS TotalSpend,
   AVG(t.Invoice_Total) AS AvgOrderValue
FROM Customers c
JOIN Transactions t ON c.Customer_ID = t.Customer_ID
GROUP BY c.Customer_ID, c.Name, c.Country;

CREATE VIEW vwCustomerValue AS
SELECT
   c.Customer_ID, c.Name, c.Country,
   COUNT(DISTINCT t.Invoice_ID) AS TotalPurchases,
   SUM(t.Invoice_Total) AS TotalSpend,
   MAX(t.Date) AS LastPurchaseDate
FROM Transactions t
JOIN Customers c ON t.Customer_ID = c.Customer_ID
GROUP BY c.Customer_ID, c.Name, c.Country;

CREATE VIEW vwCustomerSegments AS
SELECT
   Customer_ID,
   SUM(Line_Total) AS TotalSpend,
   CASE
      WHEN SUM(Line_Total) > 10000 THEN 'High Value'
      WHEN SUM(Line_Total) BETWEEN 5000 AND 10000 THEN 'Medium Value'
      ELSE 'Low Value'
   END AS Segment
FROM Transactions
GROUP BY Customer_ID;

CREATE OR ALTER VIEW vwCustomerStoreLink AS
SELECT
   c.Customer_ID, c.Name, c.City AS CustomerCity, c.Country AS CustomerCountry,
   s.Store_ID, s.Store_Name, s.City AS StoreCity, s.Country AS StoreCountry,
   s.Latitude, s.Longitude,
   COUNT(DISTINCT t.Invoice_ID) AS TotalTransactions,
   SUM(t.Line_Total) AS TotalSpend,
```

```
    AVG(t.Line_Total) AS AvgOrderValue
FROM Customers c
LEFT JOIN Stores s ON c.City = s.City AND c.Country = s.Country
LEFT JOIN Transactions t ON c.Customer_ID = t.Customer_ID
WHERE t.Transaction_Type = 'Sale'
GROUP BY c.Customer_ID, c.Name, c.City, c.Country,
    s.Store_ID, s.Store_Name, s.City, s.Country, s.Latitude, s.Longitude;

-- Product & Category Performance Views
CREATE VIEW vwTopProducts AS
SELECT
    p.Product_ID, p.Description_EN AS ProductName,
    SUM(t.Line_Total) AS TotalRevenue
FROM Transactions t
JOIN Products p ON t.Product_ID = p.Product_ID
GROUP BY p.Product_ID, p.Description_EN;

CREATE VIEW vwCategoryRevenue AS
SELECT p.Category, SUM(t.Line_Total) AS TotalRevenue
FROM Transactions t
JOIN Products p ON t.Product_ID = p.Product_ID
GROUP BY p.Category;

CREATE VIEW vwRevenueByProduct AS
SELECT
    p.Product_ID, p.Description_EN AS ProductName, p.Category, p.Sub_Category,
    SUM(t.Quantity) AS TotalUnitsSold, SUM(t.Line_Total) AS Revenue,
    AVG(t.Unit_Price) AS AvgPrice
FROM Transactions t
JOIN Products p ON t.Product_ID = p.Product_ID
GROUP BY p.Product_ID, p.Description_EN, p.Category, p.Sub_Category;

CREATE VIEW vwTop10Products AS
SELECT TOP 10
    p.Product_ID, p.Description_EN AS ProductName,
    SUM(t.Line_Total) AS TotalRevenue
FROM Transactions t
JOIN Products p ON t.Product_ID = p.Product_ID
GROUP BY p.Product_ID, p.Description_EN
ORDER BY TotalRevenue DESC;

-- Employee Performance Views
CREATE VIEW vwRevenueByEmployee AS
SELECT
    e.Employee_ID, e.Name, e.Position, s.Store_Name,
    SUM(t.Line_Total) AS RevenueGenerated,
    COUNT(DISTINCT t.Invoice_ID) AS TotalInvoicesHandled
FROM Transactions t
JOIN Employees e ON t.Employee_ID = e.Employee_ID
JOIN Stores s ON t.Store_ID = s.Store_ID
GROUP BY e.Employee_ID, e.Name, e.Position, s.Store_Name;

-- Discount Views
CREATE OR ALTER VIEW vwDiscountImpact AS
SELECT
    YEAR(t.Date) AS Year,
    COUNT(DISTINCT t.Invoice_ID) AS TotalInvoices,
```

```
    COUNT(DISTINCT t.Customer_ID) AS TotalCustomers,
    AVG(t.Discount) AS DiscountPercent,
    SUM(t.Unit_Price * t.Quantity) AS PreDiscountRevenue,
    SUM(t.Line_Total) AS RevenueAfterDiscount,
    SUM((t.Unit_Price * t.Quantity) - t.Line_Total) AS TotalDiscountAmount,
    SUM((t.Unit_Price - p.Production_Cost) * t.Quantity) AS Profit,
    AVG((t.Unit_Price - p.Production_Cost) / NULLIF(t.Unit_Price, 0) * 100) AS ProfitMargin,
    CASE
        WHEN SUM((t.Unit_Price * t.Quantity) - t.Line_Total) > 0
            THEN SUM(t.Line_Total) / SUM((t.Unit_Price * t.Quantity) - t.Line_Total)
        ELSE NULL
    END AS DiscountROI,
    (AVG(t.Discount) * 100) AS [DiscountPenetration(%)]
FROM dbo.Transactions t
JOIN dbo.Products p ON t.Product_ID = p.Product_ID
WHERE t.Transaction_Type = 'Sale'
GROUP BY YEAR(t.Date);

CREATE OR ALTER VIEW vwDiscountEffectiveness AS
SELECT
    p.Category, p.Sub_Category, d.Description, d.[Start], d.[End],
    COUNT(DISTINCT t.Invoice_ID) AS InvoicesDuringCampaign,
    SUM(t.Line_Total) AS RevenueDuringCampaign,
    AVG(t.Discount) AS DiscountPercent,
    SUM((t.Unit_Price * t.Quantity) - t.Line_Total) AS TotalDiscountAmount,
    SUM((t.Unit_Price - p.Production_Cost) * t.Quantity) AS Profit,
    AVG((t.Unit_Price - p.Production_Cost) / NULLIF(t.Unit_Price, 0) * 100) AS ProfitMargin
FROM dbo.Transactions t
JOIN dbo.Products p ON t.Product_ID = p.Product_ID
JOIN dbo.discounts d ON p.Category = d.Category AND p.Sub_Category = d.Sub_Category
WHERE t.Transaction_Type = 'Sale' AND t.Date BETWEEN d.[Start] AND d.[End]
GROUP BY p.Category, p.Sub_Category, d.Description, d.[Start], d.[End];

CREATE OR ALTER VIEW vwDiscountByCategory AS
SELECT
    p.Category,
    AVG(t.Discount) AS AvgDiscountPercent,
    SUM(t.Line_Total) AS TotalRevenue,
    CAST((SUM((t.Unit_Price - p.[Production_Cost]) * t.Quantity) * 100.0) / NULLIF(SUM(t.Line_Total), 0)
AS DECIMAL(10,2)) AS AvgProfitMarginPct
FROM dbo.Transactions t
JOIN dbo.Products p ON t.Product_ID = p.Product_ID
WHERE t.Transaction_Type = 'Sale'
GROUP BY p.Category;

-- Business KPI Views
CREATE VIEW vwRepeatCustomers AS
SELECT
    COUNT(DISTINCT CASE WHEN OrderCount > 1 THEN Customer_ID END) * 1.0 / COUNT(DISTINCT
Customer_ID) AS RepeatPurchaseRatio
FROM (SELECT Customer_ID, COUNT(DISTINCT Invoice_ID) AS OrderCount FROM Transactions GROUP
BY Customer_ID) c;

CREATE OR ALTER VIEW vwPeakSalesDay AS
SELECT
    DATENAME(WEEKDAY, t.Date) AS DayOfWeek,
    COUNT(DISTINCT t.Invoice_ID) AS Total_Transactions,
```

```
    SUM(t.Quantity) AS Total_Units_Sold,
    SUM(t.Line_Total) AS DailyRevenue,
    AVG(t.Line_Total) AS AvgOrderValue,
    COUNT(DISTINCT t.Customer_ID) AS Unique_Customers,
    COUNT(DISTINCT t.Store_ID) AS Active_Stores,
    COUNT(DISTINCT t.Employee_ID) AS Active_Employees,
    pm.Payment_Method,
    p.Category
FROM Transactions t
LEFT JOIN Products p ON t.Product_ID = p.Product_ID
LEFT JOIN (SELECT DISTINCT Invoice_ID, Payment_Method FROM Transactions) pm ON t.Invoice_ID =
pm.Invoice_ID
WHERE t.Transaction_Type = 'Sale'
GROUP BY DATENAME(WEEKDAY, t.Date), pm.Payment_Method, p.Category;

CREATE VIEW vwPaymentDistribution AS
SELECT Payment_Method, COUNT(*) AS TotalTransactions, SUM(Line_Total) AS TotalRevenue
FROM Transactions
GROUP BY Payment_Method;

CREATE VIEW vwRevenueByYearMonth AS
SELECT YEAR(Date) AS Year, MONTH(Date) AS Month, SUM(Line_Total) AS MonthlyRevenue
FROM Transactions
GROUP BY YEAR(Date), MONTH(Date);

-- Gross/Net Revenue Views
CREATE VIEW vwProductGrossRevenue AS
SELECT
    p.Product_ID, p.Description_EN AS ProductName, p.Category, p.Sub_Category,
    SUM(t.Quantity) AS TotalUnitsSold, SUM(t.Line_Total) AS GrossRevenue,
    AVG(t.Unit_Price) AS AvgPrice, AVG(p.Production_Cost) AS Cost,
    SUM(t.Line_Total) - SUM(p.Production_Cost * t.Quantity) AS GrossProfit
FROM Transactions t
JOIN Products p ON t.Product_ID = p.Product_ID
WHERE t.Transaction_Type = 'Sale'
GROUP BY p.Product_ID, p.Description_EN, p.Category, p.Sub_Category;

CREATE VIEW vwProductReturns AS
SELECT
    p.Product_ID, p.Description_EN AS ProductName, p.Category, p.Sub_Category,
    COUNT(*) AS ReturnCount, SUM(t.Line_Total) AS ReturnValue
FROM Transactions t
JOIN Products p ON t.Product_ID = p.Product_ID
WHERE t.Transaction_Type = 'Return'
GROUP BY p.Product_ID, p.Description_EN, p.Category, p.Sub_Category;

CREATE VIEW vwProductRevenueSummary AS
SELECT
    g.Product_ID, g.ProductName, g.Category, g.Sub_Category,
    g.GrossRevenue, g.GrossProfit,
    ISNULL(r.ReturnValue, 0) AS ReturnValue,
    (g.GrossRevenue + ISNULL(r.ReturnValue, 0)) AS NetRevenue,
    (g.GrossProfit + ISNULL(r.ReturnValue, 0)) AS NetProfit
FROM vwProductGrossRevenue g
LEFT JOIN vwProductReturns r ON g.Product_ID = r.Product_ID;

CREATE VIEW vwStoreGrossRevenue AS
```

```
SELECT s.Store_ID, s.Store_Name, s.Country, SUM(t.Line_Total) AS GrossRevenue
FROM Transactions t
JOIN Stores s ON t.Store_ID = s.Store_ID
WHERE t.Transaction_Type = 'Sale'
GROUP BY s.Store_ID, s.Store_Name, s.Country;

CREATE VIEW vwStoreNetRevenue AS
SELECT g.Store_ID, g.Store_Name, g.Country, g.GrossRevenue + ISNULL(r.ReturnValue, 0) AS
NetRevenue
FROM vwStoreGrossRevenue g
LEFT JOIN (SELECT Store_ID, SUM(Line_Total) AS ReturnValue FROM Transactions WHERE
Transaction_Type = 'Return' GROUP BY Store_ID) r ON g.Store_ID = r.Store_ID;

CREATE VIEW vwCategoryGrossRevenue AS
SELECT p.Category, SUM(t.Line_Total) AS GrossRevenue
FROM Transactions t
JOIN products p ON t.Product_ID = p.Product_ID
WHERE t.Transaction_Type = 'Sale'
GROUP BY p.Category;

CREATE VIEW vwCategoryReturns AS
SELECT p.Category, SUM(t.Line_Total) AS ReturnValue
FROM Transactions t
JOIN products p ON t.Product_ID = p.Product_ID
WHERE t.Transaction_Type = 'Return'
GROUP BY p.Category;

CREATE VIEW vwCategoryRevenueSummary AS
SELECT g.Category, g.GrossRevenue, ISNULL(r.ReturnValue, 0) AS ReturnValue, (g.GrossRevenue +
ISNULL(r.ReturnValue, 0)) AS NetRevenue
FROM vwCategoryGrossRevenue g
LEFT JOIN vwCategoryReturns r ON g.Category = r.Category;

CREATE VIEW vwCustomerValueNet AS
SELECT c.Customer_ID, c.Name, SUM(t.Line_Total) AS NetSpend
FROM Transactions t
JOIN Customers c ON t.Customer_ID = c.Customer_ID
GROUP BY c.Customer_ID, c.Name;

-- Advanced Views
CREATE OR ALTER VIEW vwCustomerStoreProfile AS
SELECT
  c.Customer_ID, c.Name AS CustomerName, c.Country AS CustomerCountry,
  cs.Segment AS CustomerSegment,
  ISNULL(cv.NetSpend, c.TotalSpend) AS CustomerLifetimeValue,
  CASE
    WHEN DATEDIFF(DAY, MAX(t.Date), GETDATE()) <= 180 THEN 'Active'
    WHEN DATEDIFF(DAY, MAX(t.Date), GETDATE()) BETWEEN 181 AND 365 THEN 'At_Risk'
    ELSE 'Churned'
  END AS Lifecycle_Stage,
  s.Store_ID, s.Store_Name, s.City AS StoreCity, s.Country AS StoreCountry,
  s.Latitude, s.Longitude,
  COUNT(DISTINCT t.Invoice_ID) AS TotalTransactions,
  SUM(CASE WHEN t.Transaction_Type = 'Sale' THEN t.Line_Total ELSE 0 END) AS TotalSpend,
  AVG(CASE WHEN t.Transaction_Type = 'Sale' THEN t.Line_Total ELSE NULL END) AS AvgOrderValue,
  SUM(CASE WHEN t.Transaction_Type = 'Return' THEN t.Line_Total ELSE 0 END) AS ReturnValue,
```

```
   CAST((SUM(CASE WHEN t.Transaction_Type = 'Return' THEN t.Line_Total ELSE 0 END) * -100.0) /
NULLIF(SUM(CASE WHEN t.Transaction_Type = 'Sale' THEN t.Line_Total ELSE 0 END), 0) AS
DECIMAL(10,2)) AS ReturnRatePct,
   SUM(CASE WHEN t.Transaction_Type = 'Sale' THEN t.Line_Total ELSE 0 END) + SUM(CASE WHEN
t.Transaction_Type = 'Return' THEN t.Line_Total ELSE 0 END) AS NetRevenue
FROM dbo.Transactions t
LEFT JOIN dbo.vwCustomerOverview c ON t.Customer_ID = c.Customer_ID
LEFT JOIN dbo.vwCustomerSegments cs ON t.Customer_ID = cs.Customer_ID
LEFT JOIN dbo.vwCustomerValueNet cv ON t.Customer_ID = cv.Customer_ID
LEFT JOIN dbo.Stores s ON t.Store_ID = s.Store_ID
GROUP BY c.Customer_ID, c.Name, c.Country, cs.Segment, cv.NetSpend, c.TotalSpend,
   s.Store_ID, s.Store_Name, s.City, s.Country, s.Latitude, s.Longitude;

-- Payment Views
CREATE OR ALTER VIEW vwPaymentMethodRevenue AS
SELECT Payment_Method, COUNT(DISTINCT Invoice_ID) AS Total_Transactions, SUM(Line_Total) AS
Total_Revenue, AVG(Line_Total) AS Avg_Transaction_Value
FROM Transactions
WHERE Transaction_Type = 'Sale'
GROUP BY Payment_Method;

CREATE OR ALTER VIEW vwPaymentMethodMonthlyTrend AS
SELECT FORMAT(Date, 'yyyy-MM') AS YearMonth, Payment_Method, SUM(Line_Total) AS
Monthly_Revenue, COUNT(DISTINCT Invoice_ID) AS Monthly_Transactions
FROM Transactions
WHERE Transaction_Type = 'Sale'
GROUP BY FORMAT(Date, 'yyyy-MM'), Payment_Method;

CREATE OR ALTER VIEW vwPaymentMethodBySegment AS
SELECT c.Customer_ID, s.Segment, t.Payment_Method, SUM(t.Line_Total) AS Total_Revenue,
COUNT(DISTINCT t.Invoice_ID) AS Num_Transactions
FROM Transactions t
JOIN vwCustomerSegments s ON t.Customer_ID = s.Customer_ID
JOIN Customers c ON t.Customer_ID = c.Customer_ID
WHERE t.Transaction_Type = 'Sale'
GROUP BY c.Customer_ID, s.Segment, t.Payment_Method;

CREATE OR ALTER VIEW vwPaymentShareSummary AS
SELECT Payment_Method, SUM(Line_Total) AS Total_Revenue, SUM(Line_Total) * 100.0 /
SUM(SUM(Line_Total)) OVER() AS Revenue_Share_Percent
FROM Transactions
WHERE Transaction_Type = 'Sale'
GROUP BY Payment_Method;

CREATE OR ALTER VIEW vwPaymentNetRevenue AS
SELECT Payment_Method,
   SUM(CASE WHEN Transaction_Type = 'Sale' THEN Line_Total ELSE 0 END) AS Gross_Revenue,
   SUM(CASE WHEN Transaction_Type = 'Return' THEN Line_Total ELSE 0 END) AS Return_Value,
   SUM(Line_Total) AS Net_Revenue,
   (SUM(CASE WHEN Transaction_Type = 'Return' THEN Line_Total ELSE 0 END) * 100.0 /
NULLIF(SUM(CASE WHEN Transaction_Type = 'Sale' THEN Line_Total ELSE 0 END), 0)) AS
Return_Rate_Percent
FROM Transactions
GROUP BY Payment_Method;

CREATE OR ALTER VIEW vwPaymentNetMonthlyTrend AS
SELECT FORMAT(Date, 'yyyy-MM') AS YearMonth, Payment_Method,
```

```
   SUM(CASE WHEN Transaction_Type = 'Sale' THEN Line_Total ELSE 0 END) AS Gross_Revenue,
   SUM(CASE WHEN Transaction_Type = 'Return' THEN Line_Total ELSE 0 END) AS Return_Value,
   SUM(Line_Total) AS Net_Revenue,
   (SUM(CASE WHEN Transaction_Type = 'Return' THEN Line_Total ELSE 0 END) * 100.0 /
NULLIF(SUM(CASE WHEN Transaction_Type = 'Sale' THEN Line_Total ELSE 0 END), 0)) AS
Return_Rate_Percent
FROM Transactions
GROUP BY FORMAT(Date, 'yyyy-MM'), Payment_Method;

-- Returns Views
CREATE VIEW vwReturnsSummary AS
SELECT t.Transaction_Type, COUNT(*) AS TotalReturns, SUM(t.Line_Total) AS TotalReturnValue,
   COUNT(DISTINCT t.Customer_ID) AS UniqueCustomers, COUNT(DISTINCT t.Product_ID) AS
UniqueProducts,
   COUNT(DISTINCT t.Store_ID) AS StoresAffected
FROM Transactions t
WHERE t.Transaction_Type = 'Return'
GROUP BY t.Transaction_Type;

CREATE VIEW vwReturnsByCategory AS
SELECT p.Category, COUNT(*) AS TotalReturns, SUM(t.Line_Total) AS TotalReturnValue
FROM Transactions t
JOIN products p ON t.Product_ID = p.Product_ID
WHERE t.Transaction_Type = 'Return'
GROUP BY p.Category;

CREATE VIEW vwReturnsByCountry AS
SELECT s.Country, COUNT(*) AS TotalReturns, SUM(t.Line_Total) AS TotalReturnValue
FROM Transactions t
JOIN stores s ON t.Store_ID = s.Store_ID
WHERE t.Transaction_Type = 'Return'
GROUP BY s.Country;
```