The Product Company

~ Final Data Mart Development Report ~

Team # 2

Team Members

Sailee Rumao

Swati

Sanya Kaur Gulati

Date

05/01/2019

ISTE-DW Data Warehousin

~ Table of Contents ~

I. Data Mart Design Definition

1. Universe of Discourse

This data mart is an integration of sales data from TPC-E, TPC-W and PEC.

The data mart would allow the end user to investigate the key performance measures like gross profit, sale amount, sale quantity and number of days to ship for the products being sold to the customers across all divisions so as to effectively manage the financial performance of the product company. The performance can be analyzed on yearly, quarterly, monthly and daily basis.

2. Information Package

Process Name: Financial Performance

Grain: A sale or purchase transaction made by the customers for any product on **daily basis** in any of the three divisions of the company - TPCE, PEC and TPCW is the grain.

Customer DIM	**Product DIM**	**SaleDate DIM**	**OrderDate DIM**	**Shipping_payment_order_junk DIM**
Customer_SK	Product_SK	SalesDate_SK	OrderDate_SK	Payment_Order_Shipping_Junk_SK
CustID	ProductID	SalesDate	OrderDate	ShippingMethod
CustomerName	ProductName	SalesYear	OrderYear	PaymentMethod

Add1	Price1	SalesQuarter	OrderQuarter	OrderMethod
Addr2	Price2	SalesMonth	OrderMonth	
City	Unit Cost	SalesWeek	OrderWeek	
State	ProductTypeID	SalesDay	OrderDay	
Zip	ProductTypeDescription	DayofWeek	DayofWeek	
CustTypeID	SupplierID	SalesFiscal Year	OrderFiscalYear	
TypeName	SupplierName	SalesFiscal Quarter	OrderFiscalQuarter	
DivisionID	SupplierAddr1	SalesFiscal Month	OrderFiscalMonth	
	SupplierAddr2	SalesFiscal Week	OrderFiscalWeek	
	SupplierCity			
	SupplierState			
	SupplierZip			
	BUID			
	BUName			
	BUAbbrev			

Facts: Profit, Amount, Quantity, ShipCost, Discounted, Number of days to Ship

3. Entity Definitions

Entity	**Entity Definition** (*genus differentia*)
Customer	**Def**: This dimension contains information about the customers who buy products from the company. **Attributes**: 1. **Customer_SK**: It is the surrogate key of the customer dimension. 2. **CustID:** It is the ID which is unique to each customer. It is also the natural key. 3. **CustomerName:** The name of the customer. 4. **Addr1:** The street address of the customer. 5. **Addr2:** The details of the address like P.O. Box number, Department Number, Suite Number etc. 6. **City:** The city in which the customer lives. 7. **State:** The state in which the customer lives. 8. **Zip:** The 5-digit zip code in which the customer lives. 9. **CustTypeID:** The ID of the type of the customer. It has 4 values: S (State/Local Govt), E (Education), F (US Govt) and C (Commercial).

	10. **TypeName:** The category of the type of the customer, i.e. Commercial, Education, State/Local Govt and US Govt. 11. **DivisionID:** The ID associated with each division of the product company. It has 3 values: 1(TPCE), 2(TPCW) and 3(PEC).
Product	**Def**: This dimension contains information about the products sold or handled by the three divisions - TPCE, TPCW and PEC. **Attributes**: 1. **Product_SK:** It is the surrogate key of the product dimension. 2. **ProductID:** It is the ID which is unique to each product. It is also the natural key. 3. **ProductName:** The name of the product. 4. **Price1:** The original price of the product. 5. **Price 2:** The price of the product after discount. 6. **UnitCost:** The cost of the product per unit for each division. 7. **ProductTypeID:** The ID that represents the type of product 8. **ProductTypeDescription:** The descriptions about the types of product in The Product Company. 9. **SupplierID:** The ID of the supplier.

	10. **SupplierName:** The name or description of the supplier that is providing the product. 11. **SupplierAddr1:** The street address of the supplier like 1616 Goggles Drive, 1618 Cookbook Circle, Greenland Street etc. 12. **SupplierAddr2:** The name of the person the delivery is address to. 13. **SupplierCity:** The city in which the supplier resides. 14. **SupplierState:** The state in which the supplier resides. 15. **SupplierZip:** The zip code in which the supplier resides. 16. **BUID:** The ID of the business unit 17. **BUName:** The name of the business unit 18. **BUAbbrev:** The abbreviation of the Business Units. 19. **DivisionID:** The ID associated with each division of the product company. It has 3 values: 1(TPCE), 2(TPCW) and 3(PEC).
SaleDate	**Def**: The SaleDate dimension contains details about the date on which the sale was made for all of the three divisions - TPCE, TPCW and PEC. **Attributes**: 1. **SalesDate_SK:** It is the surrogate key of the SaleDate dimension.

	2. **SalesDate**: The date on which the sale transaction was made. 3. **SalesYear**: The calendar year in which the sale transaction was made. 4. **SalesQuarter**: The calendar quarter in which the sale transaction was made. 5. **SalesMonth**: The calendar month in which the sale transaction was made. 6. **SalesWeek**: The calendar week in which the sale transaction was made. 7. **SalesDay**: The calendar day on which the sale transaction was made. 8. **DayOfWeek**: The day of the week on which the sale transaction was made. 9. **SalesFiscalYear**: The fiscal year in which the sale transaction was made. 10. **SalesFiscalQuarter**: The fiscal quarter in which the sale transaction was made. 11. **SalesFiscalMonth:** The fiscal month in which the sale transaction was made. 12. **SalesFiscalWeek:** The fiscal week in which the sale transaction was made.

OrderDate	**Def**: The OrderDate dimension contains details about the date on which the products were ordered by the customer of PEC. **Attributes**: 1. **OrderDate_SK:** It is the surrogate key of the OrderDate dimension. 2. **OrderDate**: The date on which the products were ordered by the customer. 3. **OrderYear**: The calendar year in which the products were ordered by the customer. 4. **OrderQuarter**: The calendar quarter in which the products were ordered by the customer. 5. **OrderMonth**: The calendar month in which the products were ordered by the customer. 6. **OrderWeek**: The calendar week in which the products were ordered by the customer. 7. **OrderDay**: The calendar day on which the products were ordered by the customer. 8. **DayOfWeek**: The day of the week in which the products were ordered by the customer. 9. **OrderFiscalYear**: The fiscal year in which the products were ordered by the customer. 10. **OrderFiscalQuarter**: The fiscal quarter in which the products were ordered by the customer.

	11. **OrderFiscalMonth:** The fiscal month in which the products were ordered by the customer. 12. **OrderFiscalWeek:** The fiscal week in which the products were ordered by the customer.
Shipping_payment_order_junk	**Def**: The junk dimension contains some details about each sales transaction that do not belong to any other entity. **Attributes**: 1. **Payment_Order_Shipping_JunkSK**: It is the surrogate key of the junk dimension. 2. **ShippingMethod**: It states the method of shipping used by the customer. Train, Truck and Air are the three possible shipping methods. 3. **PaymentMethod**: It states the method of payment used by the customer. Cod, Cash and Charge are the three possible shipping methods. 4. **OrderMethod:** It states the method of ordering used by the customer. Email, Internet and Phone are the three possible ordering methods.
Sales_Fact	**Def**: This is the fact table. Each row of the fact table represents a sale transaction made in any of the three divisions: TPCE, TPCW and PEC.

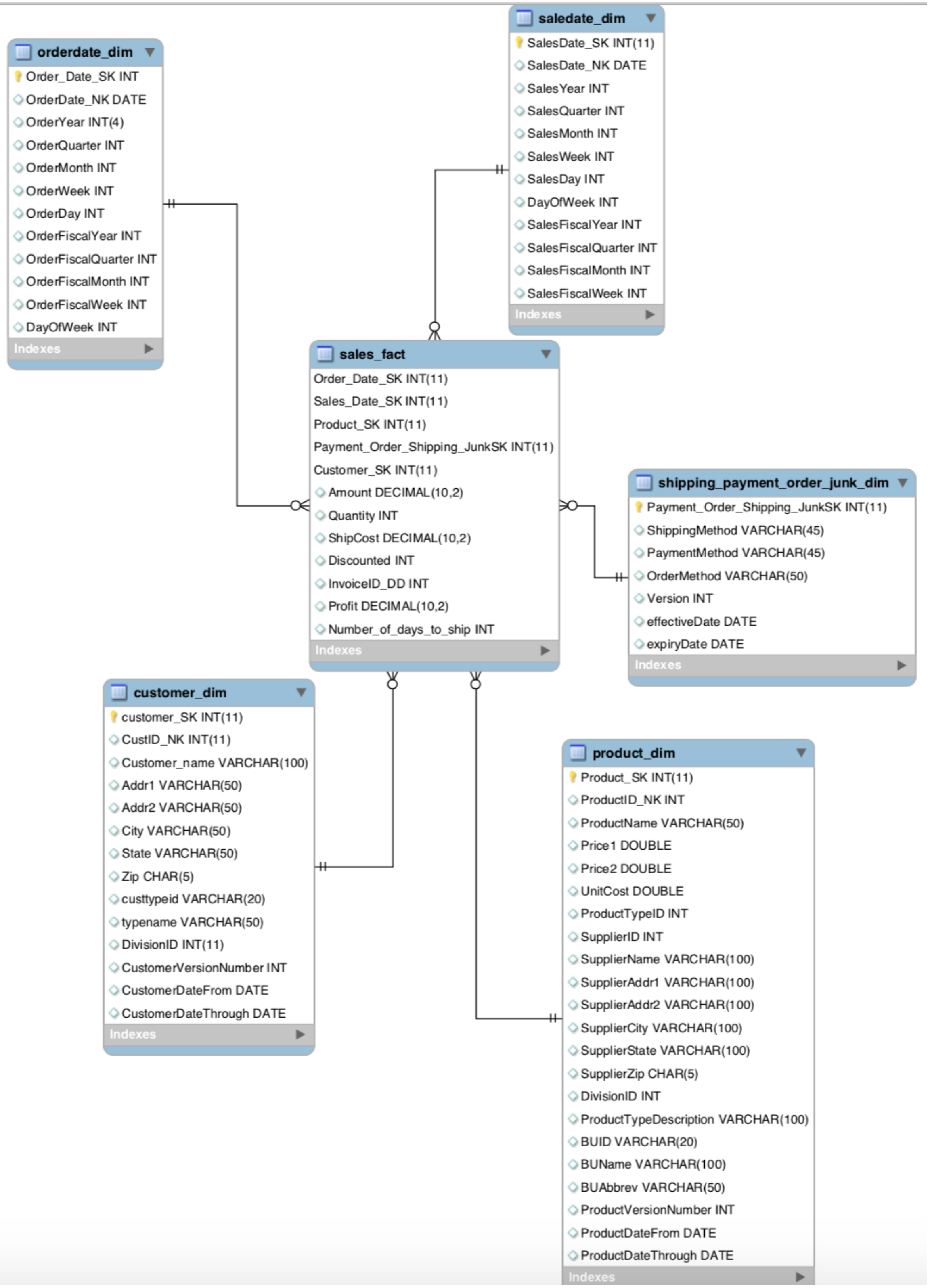
	<u>**Attributes**</u>: 1. **Customer_SK**: A part of the composite primary key of the Fact table. It is also a foreign key and is used to fetch information from the customer dimension. 2. **Product_SK**: A part of the composite primary key of the Fact table. It is also a foreign key and is used to fetch information from the product dimension. 3. **Sales_Date_SK**: A part of the composite primary key of the Fact table. It is also a foreign key and is used to fetch information from the saleDate dimension. 4. **Order_Date_SK:** attribute is a part of the composite primary key of the Fact table. It is also a foreign key and is used to fetch information from the orderDate dimension. 5. **Shipping_Payment_Order_Junk_SK**: A part of the composite primary key of the Fact table. It is also a foreign key and is used to fetch information from the shipping_payment_order_junk dimension. 6. **Amount**: Total sale price for each invoice. The amount varies depending upon whether the item is discounted or not. 7. **Quantity**: Total number of products associated with a particular invoice.

	8. **Discounted**: Depicts whether the product purchased is discounted or not. It has values 0 and 1 with 0 depicting “not discounted” and 1 depicting “discounted”. 9. **Profit**: The profit made by the company. It has been calculated using the formula : Amount - (UnitCost * Quantity). 10. **Number_of_days_to_ship**: The number of days it took to deliver the order from the date on which the order was placed. The formula used: (SalesDate - OrderDate) 11. **InvoiceID_DD**: This is a degenerate dimension. It contains Invoice ID of each of the sales transaction.

II. Dimensional Model

Dimensional Model:

- Each dimension has been denormalized. All the entities, attributes, relationships and cardinalities have been mentioned in the Crow Foot notation format.
- All the dimensions have "_dim" in the name and the fact has "_fac" in the name so that they can be easily differentiated.
- The dimensions are being connected with the fact table with help of surrogate keys, which behaves as the primary key along with optional cardinality (zero or many).
- Shipping_payment_order_junk is a junk dimension of the model and contains attributes like the order method, the payment method and the shipping method.
- InvoiceID_DD is a degenerate dimension of the dimensional model.
- Amount, Quantity, Profit, ShipCost, Number_of_days_to_ship and Discounted are the facts. Profit is fully additive and Number_of_days_to_ship is semi additive.



orderdate_dim
Order_Date_SK INT
OrderDate_NK DATE
OrderYear INT(4)
OrderQuarter INT
OrderMonth INT
OrderWeek INT
OrderDay INT
OrderFiscalYear INT
OrderFiscalQuarter INT
OrderFiscalMonth INT
OrderFiscalWeek INT
DayOfWeek INT
Indexes
saledate_dim
SalesDate_SK INT(11)
SalesDate_NK DATE
SalesYear INT
SalesQuarter INT
SalesMonth INT
SalesWeek INT
SalesDay INT
DayOfWeek INT
SalesFiscalYear INT
SalesFiscalQuarter INT
SalesFiscalMonth INT
SalesFiscalWeek INT
Indexes
sales_fact
Order_Date_SK INT(11)
Sales_Date_SK INT(11)
Product_SK INT(11)
Payment_Order_Shipping_JunkSK INT(11)
Customer_SK INT(11)
Amount DECIMAL(10,2)
Quantity INT
ShipCost DECIMAL(10,2)
Discounted INT
InvoiceID_DD INT
Profit DECIMAL(10,2)
Number_of_days_to_ship INT
Indexes
shipping_payment_order_junk_dim
Payment_Order_Shipping_JunkSK INT(11)
ShippingMethod VARCHAR(45)
PaymentMethod VARCHAR(45)
OrderMethod VARCHAR(50)
Version INT
effectiveDate DATE
expiryDate DATE
Indexes
customer_dim
customer_SK INT(11)
CustID_NK INT(11)
Customer_name VARCHAR(100)
Addr1 VARCHAR(50)
Addr2 VARCHAR(50)
City VARCHAR(50)
State VARCHAR(50)
Zip CHAR(5)
custtypeid VARCHAR(20)
typename VARCHAR(50)
DivisionID INT(11)
CustomerVersionNumber INT
CustomerDateFrom DATE
CustomerDateThrough DATE
Indexes
product_dim
Product_SK INT(11)
ProductID_NK INT
ProductName VARCHAR(50)
Price1 DOUBLE
Price2 DOUBLE
UnitCost DOUBLE
ProductTypeID INT
SupplierID INT
SupplierName VARCHAR(100)
SupplierAddr1 VARCHAR(100)
SupplierAddr2 VARCHAR(100)
SupplierCity VARCHAR(100)
SupplierState VARCHAR(100)
SupplierZip CHAR(5)
DivisionID INT
ProductTypeDescription VARCHAR(100)
BUID VARCHAR(20)
BUName VARCHAR(100)
BUAbbrev VARCHAR(50)
ProductVersionNumber INT
ProductDateFrom DATE
ProductDateThrough DATE
Indexes

III. Data Staging: ETL – Data Extract File Definitions

TPC-W: 6 CSV Files Provided

Business Unit	**File Name**	**Format**	**Attributes**
TPC-W	TPCWbusiness_unit.csv	Fields enclosed in double quotes (“”) and separated by semicolon (;)	BUID - String NAME- String ABBREV-String
TPC-W	TPCWCustomer.csv	Fields enclosed in double quotes (“”) and separated by semicolon (;)	custID-Integer name-String address-String city-String state-String zip-Integer custType-String
TPC-W	TPCWcustomer_type.csv	Fields enclosed in double quotes (“”) and separated by semicolon (;)	CUSTTYPEID-String TYPENAME-String

TPC-W	TPCWinvoice.csv	Fields separated by comma (,)	Invoice-Integer custID-Integer prodID-Integer salesDate-String amt-Integer qty-Integer discounted-Integer
TPC-W	TPCWproduct.csv	Fields enclosed in double quotes (“”) and separated by semicolon (;). Rows enclosed by double quotes (“”)	ProductID- Integer ProductName- String Price1- Number Price2- Number Unit Cost- Number Supplier Name- String Supplier Address- String Supplier city- String Supplier State- String Supplier zipcode- String Product Type ID- Integer

TPC-W	TPCWproduct_type.csv	Fields enclosed in double quotes (“”) and separated by semicolon (;). Rows enclosed by double quotes (“”)	PRODTYPEID- String TYPEDESCRIPTION-String BUID- String

PEC: 7 CSV Files Provided

Business Unit	**File Name**	**Format**	**Attributes**
PEC	PECbusiness_unit.csv	Fields enclosed in double quotes (“”) and separated by semicolon (;)	BUID - String NAME- String ABBREV-String
PEC	PECcustomer.csv	Fields separated by semicolon (;)	custID-Integer name-String address-String city-String state-String zip-Integer

			custType-String
PEC	PECcustomer_type.csv	Fields enclosed in double quotes (“”) and separated by semicolon (;)	CUSTTYPEID-String TYPENAME-String
PEC	PECinvoice.csv	Fields separated by comma (,)	Invoice-Integer Cust-ID-Integer salesDate-Date prodid-integer amt-Integer qty-Integer shipMethod-String shipCost-Decimal paymentMethod-String orderMethod-String

			orderDate-Date discounted-Integer
PEC	PECmanufacturingCosts.csv	Fields separated by pipe (|)	Year- Integer Month- Integer ProdID- Integer manufacturingCost- Integer

PEC	PECproduct_type.csv	Fields enclosed in double quotes (“”) and separated by semicolon (;). Rows enclosed by double quotes (“”)	PRODTYPEID- String TYPEDESCRIPTION- String BUID- String
PEC	PECproduct.csv	Fields enclosed in double quotes (“”) and separated by semicolon (;). Rows enclosed by double quotes (“”)	prodid- Integer prodDescription- String price1- Decimal price2- Decimal unitCost- Decimal supplierName- String productTypeID- Integer

TPC-E: 8 CSV Files Provided

Business Unit	**File Name**	**Format**	**Attributes**

TPC-E	business_unit.csv	Fields enclosed in double quotes (“”) and separated by semicolon (;)	BUID - String NAME- String ABBREV-String
TPC-E	customer.csv	Fields enclosed in double quotes (“”) and separated by semicolon (;)	CUSTID-Integer NAME-String ADDR1-String ADDR2- String CITY-String STATE-String ZIP-Integer CUSTTYPEID-String
TPC-E	customer_type.csv	Fields enclosed in double quotes (“”) and separated by semicolon (;)	CUSTTYPEID-String TYPENAME-String
TPC-E	invoice.csv	Fields separated by comma (,)	InvoiceID-Integer custID-Integer salesDate-Date

TPC-E	invoice_details.csv	Fields separated by comma (,)	InvoiceID – Integer prodID- Integer amt- Decimal qty- Integer discounted-Integer
TPC-E	prod_type.csv	Fields enclosed in double quotes (“”) and separated by semicolon (;).	PRODTYPEID- String TYPEDESCRIPTION- String BUID- String
TPC-E	product.csv	Fields enclosed in double quotes (“”) and separated by semicolon (;).	ProductID- Integer ProductName- String Price1- Number Price2- Number Unit Cost- Number Supplier Name- String Supplier Address- String Supplier city- String Supplier State- String Supplier zipcode- String Product Type ID- Integer

TPC-E	supplier.csv	Fields enclosed in double quotes (“”) and separated by semicolon (;).	SUPPLIERID-Integer NAME- String ADDR1- String ADDR2- String CITY- String STATE- String ZIP- Integer

IV. Data Staging: ETL – Source-to-Target Mappings

Follow the same format as indicated in “The Data Warehouse ETL Toolkit” by Kimball & Caserta, Fig. 3.1 on page 60. This is available on Books 24x7. The table should be in alphabetical order table name and column name.

Target					**Source**				
Target Table Name	**Target Column Name**	**Data Type**	**Table Type**	**SCD**	**Source Database**	**Source Table Name**	**Source Column Name**	**Data Type**	**Transformation**
product_dim	Product_SK	INT	Dimension	0					Refer KTR Screenshot in appendix file for all transformations.

	ProductID_NK	INT		0	TPCW,TPCE,PEC	product,TPCWproduct,PECproduct	PRODID,prodid	String	Changed data type to INT
	ProductName	VARCHAR(50)	Dimension	1	TPCW,TPCE,PEC	product,TPCWproduct,PECproduct	prodDescription,DESCRIPTION	String	Changed attribute name to ProductName
	Price1	DOUBLE	Dimension	0	TPCW,TPCE,PEC	product,TPCWproduct,PECproduct	price1,PRICE1	String	Changed the attribute name to Price. Changed the datatype to INT
	Price2	DOUBLE	Dimension	0	TPCW,TPCE,PEC	product,TPCWproduct,PECproduct	price2,PRICE2	String	Changed the attribute name to Price2

									Changed the datatype to INT
	UnitCost	DOUBLE	Dimension	0	TPCW,TPCE,PEC	product,TPCWproduct,PECproduct	unitCost,UNITCOST	String	Changed the attribute name to UnitCost Changed datatype to INT
	ProductTypeID	INT	Dimension	1	TPCW,TPCE,PEC	prod_type,TPCWproduct_type,PECproduc_typet	PRODTYPEID	String	Merged with product_dim for each division.
	SupplierID	INT	Dimension	1	TPCE	supplier	SUPPLIERID	string	Changed the attribute name to SupplierID and merged

									in the product_dim table for each division.
	SupplierName	VARCHAR(100)	Dimension	1	TPCE,PEC	supplier	NAME	string	Changed the attribute name to SupplierName and merged in the product_dim table for each division
	SupplierAddr1	VARCHAR(100)	Dimension	2	TPCE	supplier	ADDR1	string	Changed the attribute name to SupplierAddr1 and merged in the product_dimt

									table for each division
	SupplierAddr2	VARCHAR(100)	Dimension	2	TPCE	supplier	ADDR2	string	Changed the attribute name to SupplierAddr2 and merged in the produc_dim table for each division
	SupplierCity	VARCHAR(100)	Dimension	2	TPCE	supplier	CITY	string	Changed the attribute name to SupplierCity and merged in the produc_dim table for each division

	SupplierState	VARCHAR(100)	Dimension	2	TPCE	supplier	STATE	string	Changed the attribute name to SupplierState and merged in the produc_dim table for each division
	SupplierZip	CHAR(5)	Dimension	2	TPCE	supplier	ZIP	string	Changed the attribute name to SupplierZip and merged in the produc_dim table for each division
	DivisionID	INT	Dimension	0	TPCW,TPCE,PEC	product			Created with Pentaho 1 - TPCE 2 -TPCW

									3 – PEC
	ProductType Description	VARCHAR(100)	Dimension	0	TPCW,TPCE,PEC	prod_type,TPCWproduct_type,PECproduc_typet	TYPE DESCRIPTION	String	Changed the attribute name to ProductType Desction and merged in the product_dim table.
	BUID	VARCHAR(20)	Dimension	2	TPCW,TPCE,PEC	business_unit,PECbusiness_unit,TPCWbusiness_unit	BUID	String	First Merged to ProductType and then to Product_dim table.
	BUName	VARCHAR(100)	Dimension	2	TPCW,TPCE,PEC	business_unit,PECbusiness_unit,TPC	NAME	String	First Merged to ProductType and then to

						Wbusiness_unit			Product_dim table.
	BUAbbrev	VARCHAR(50)	Dimension	2	TPCW,TPCE,PEC	business_unit,PECbusiness_unit,TPCWbusiness_unit	ABBREV	String	First Merged to ProductType and then to Product_dim table.
customer_dim	Customer_SK	INT	Dimension	0					Refer KTR Screenshot in appendix for all transformations.
	CustID_NK	INT	Dimension	0	TPCE, TPCW, PEC	customer.csv, TPCWcustomer.csv, PECcustomer.csv	CUSTID,CustID,CustID	Numeric, Numeric, Numeric, Numeric	Changed attribute name to **custID for TPCE** as per our standardizati

									on. Rest from input files.
	CustomerName	VARCHAR(100)	Dimension	1	TPCE, TPCW, PEC	customer.csv, TPCWcustomer.csv, PECcustomer.csv	NAME,name,name	String. String. String	Changed attribute name to **CustomerName for all divisions** as per our standardization. Rest from input files.
	Addr1	VARCHAR(50)	Dimension	2	TPCE, TPCW, PEC	customer.csv, TPCWcustomer.csv,	ADDR2, Address,Address	String. String. String	Split the Address field into addr1 and addr2 using pentaho.

						PECcustomer.csv			
	Addr2	VARCHAR(50)	Dimension	2	TPCE, TPCW, PEC	customer.csv, TPCWcustomer.csv, PECcustomer.csv	ADDR1, Address,Address	String. String. String	Split the Address field into addr1 and addr2 using pentaho.
	City	VARCHAR(50)	Dimension	2	TPCE, TPCW, PEC	customer.csv, TPCWcustomer.csv, PECcustomer.csv	CITY, City,City	String. String. String	Changed attribute name to **city for TPCE** as per our standardization. Rest from input files.

	State	VARCHAR(50)	Dimension	2	TPCE, TPCW, PEC	customer.csv, TPCWcustomer.csv, PECcustomer.csv	STATE,state,state	String. String. String	Changed attribute name to **state for TPCE** as per our standardization. Rest from input files.
	Zip	CHAR(5)	Dimension	2	TPCE, TPCW, PEC	customer.csv, TPCWcustomer.csv, PECcustomer.csv	Zip,zip,zip	Numeric, Numeric, Numeric	
	CustTypeID	VARCHAR(20)	Dimension	1	TPCE, TPCW, PEC	Custome.csv, TPCWcustomer_type.csv,	CUSTTYPEID,	String. String. String	Join with customer based on **custTypeID**

						PECcustomer_type.csv	CUSTTYPEID, CUSTTYPEID		
	TypeName	VARCHAR(20)	Dimension	1	TPCE, TPCW, PEC	Customer_type.csv, TPCWcustomer_type.csv, PECcustomer_type.csv	TYPENAME, TYPENAME, TYPENAME	String. String. String	Join with customer based on **custTypeID.**
	DivisionID	INT	Dimension	0	TPCE, TPCW, PEC	customer.csv, TPCWcustomer.csv, PECcustomer.csv			Created with Pentaho 1 - TPCE 2 -TPCW 3 – PEC

orderdate_dim	Order_Date_SK	INT	Dimension	0					Used sequence in pentaho to add key. Refer KTR Screenshot in appendix for all transformations.
	OrderDate	Date	Dimension	1	PEC	PECinvoice.csv	order Date	string	Standardize date format in MM/DD/YYYY in Pentaho.
	OrderYear	INT	Dimension	1	PEC	PECinvoice.csv	order Date	string	Extracted calendar year from date in Pentaho

	OrderQuarter	INT	Dimension	1	PEC	PECinvoice.csv	order Date	string	Extracted calendar quarter from date in Pentaho
	OrderMonth	INT	Dimension	1	PEC	PECinvoice.csv	order Date	string	Extracted calendar month from date in Pentaho
	OrderWeek	INT	Dimension	1	PEC	PECinvoice.csv	order Date	string	Extracted calendar week from date in Pentaho
	OrderDay	INT	Dimension	1	PEC	PECinvoice.csv	order Date	string	Extracted calendar day from date in Pentaho
	OrderFiscalYear	INT	Dimensi	1	PEC	PECinvoice.csv	order Date	string	Extracted fiscal year

			on						from date in Pentaho
	OrderFiscalQuarter	INT	Dimension	1	PEC	PECinvoice.csv	order Date	string	Extracted fiscal quarter from date in Pentaho.
	OrderFiscal Month	INT	Dimension	1	PEC	PECinvoice.csv	order Date	string	Extracedt fiscal month from date in Pentaho.
	OrderFiscal Week	INT	Dimension	1	PEC	PECinvoice.csv	order Date	string	Extracted fiscal week from date in Pentaho.
	DayOfWeek	INT	Dimension	1	PEC	PECinvoice.csv	order Date	string	Extracted day of week from date in Pentaho.
salesdate_dim	Sales_Date_SK	INT	Dimension	0					Used sequence in pentaho to

									add key. Refer KTR Screenshot in appendix for all transformations.
	SalesDate	Date	Dimension	1	TPCE, TPCW, PEC	PECinvoice.csv,TPCWinvoice.scv,TPCEinvoice.csv	salesDate, salesDate, salesDate	string, string, string	Standardize date format in MM/DD/YYYY in Pentaho.
	SalesYear	INT	Dimension	1	TPCE, TPCW, PEC	PECinvoice.csv,TPCWinvoice.scv,TPCEinvoice.csv	salesDate, salesDate, salesDate	string, string, string	Extracted calendar year from date in Pentaho

	SalesQuarter	INT	Dimension	1	TPCE, TPCW, PEC	PECinvoice.csv,TPCWinvoice.scv,TPCEinvoice.csv	salesDate, salesDate, salesDate	string, string, string	Extracted calendar quarter from date in Pentaho
	SalesMonth	INT	Dimension	1	TPCE, TPCW, PEC	PECinvoice.csv,TPCWinvoice.scv,TPCEinvoice.csv	salesDate, salesDate, salesDate	string, string, string	Extracted calendar month from date in Pentaho
	SalesWeek	INT	Dimension	1	TPCE, TPCW, PEC	PECinvoice.csv,TPCWinvoice.scv,TPCEinvoice.csv	salesDate, salesDate, salesDate	string, string, string	Extracted calendar week from date in Pentaho
	SalesDay	INT	Dimension	1	TPCE, TPC	PECinvoice.csv,TPCWinvoi	salesDate, salesD	string, string, string	Extracted calendar day

					W, PEC	ce.scv,TPCEinvoice.csv	ate, salesDate		from date in Pentaho
	SalesFiscalYear	INT	Dimension	1	TPCE, TPCW, PEC	PECinvoice.csv,TPCWinvoice.scv,TPCEinvoice.csv	salesDate, salesDate, salesDate	string, string, string	Extracted fiscal year from date in Pentaho
	SalesFiscalQuarter	INT	Dimension	1	TPCE, TPCW, PEC	PECinvoice.csv,TPCWinvoice.scv,TPCEinvoice.csv	salesDate, salesDate, salesDate	string, string, string	Extracted fiscal quarter from date in Pentaho
	SalesFiscalMonth	INT	Dimension	1	TPCE, TPCW, PEC	PECinvoice.csv,TPCWinvoice.scv,TPCEinvoice.csv	salesDate, salesDate, salesDate	string, string, string	Extracted fiscal month from date in Pentaho

	SalesFiscalWeek	INT	Dimension	1	TPCE, TPCW, PEC	PECinvoice.csv,TPCWinvoice.scv,TPCEinvoice.csv	salesDate, salesDate, salesDate	string, string, string	Extracted fiscal week from date in Pentaho
	DayOfWeek	INT	Dimension	1	TPCE, TPCW, PEC	PECinvoice.csv,TPCWinvoice.scv,TPCEinvoice.csv	salesDate, salesDate, salesDate	string, string, string	Extracted calendar day of the week from date in Pentaho
payment_order_shipping_junk_dim	Payment_Order_Shipping_JunkSK	INT	Dimension	0	PEC				Used sequence in pentaho to add key. Refer KTR Screenshot in appendix for all transformations.

	PaymentMethod	VARCHAR(100)	Dimension	1	PEC	PECinvoice	paymentMethod	string	Combine and create cartesian product.
	ShippingMethod	VARCHAR(100)	Dimension	1	PEC	PECinvoice	shippingMethod	string	Combine and create cartesian product.
	OrderMethod	VARCHAR(100)	Dimension	1	PEC	PECinvoice	orderMethod	string	Combine and create cartesian product.
sales_fact	Order_Date_SK	INT	Fact	NA					Foreign_FK
	Sales_Date_SK	INT	Fact	NA					Foreign_FK
	Product_SK	INT	Fact	NA					Foreign_FK

	Payment_Order_Shipping_JunkSK	INT	Fact	NA					Foreign_FK
	Customer_SK	INT	Fact	NA					Foreign_FK
	Amount	DECIMAL(10,2)	Fact	NA					
	Quantity	INT	Fact	NA					
	ShipCost	DECIMAL(10,2)	Fact	NA					
	Discounted	INT	Fact	NA					
	Profit	DECIMAL(10,2)	Fact	NA					

	Number_of_days_to_ship	INT	Fact	NA					
	InvoiceID_DD	INT	Fact	NA					

V. SQL Code – Tables & Constraints

Customer DIM Creation and Constraints

```
DROP TABLE IF EXISTS `customer_dim`;

/*!40101 SET @saved_cs_client     = @@character_set_client */;

/*!40101 SET character_set_client = utf8 */;

CREATE TABLE `customer_dim` (

  `Customer_SK` int(11) NOT NULL,

  `CustID` int(11) DEFAULT NULL,

  `CustomerName` varchar(100) DEFAULT NULL,

  `Addr1` varchar(50) DEFAULT NULL,

  `Addr2` varchar(50) DEFAULT NULL,

  `City` varchar(50) DEFAULT NULL,

  `State` varchar(50) DEFAULT NULL,

  `Zip` char(5) DEFAULT NULL,

  `custtypeid` varchar(20) DEFAULT NULL,

  `typename` varchar(50) DEFAULT NULL,

  `DivisionID` int(11) DEFAULT NULL,

  PRIMARY KEY (`Customer_SK`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

/*!40101 SET character_set_client = @saved_cs_client */;
```

Order Date DIM Creation and Constraints

```
DROP TABLE IF EXISTS `orderdate_dim`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `orderdate_dim` (
  `Order_Date_SK` int(11) NOT NULL,
  `OrderDate` date DEFAULT NULL,
  `OrderYear` int(4) DEFAULT NULL,
  `OrderQuarter` int(11) DEFAULT NULL,
  `OrderMonth` int(11) DEFAULT NULL,
  `OrderWeek` int(11) DEFAULT NULL,
  `OrderDay` int(11) DEFAULT NULL,
  `OrderFiscalYear` int(11) DEFAULT NULL,
  `OrderFiscalQuarter` int(11) DEFAULT NULL,
  `OrderFiscalMonth` int(11) DEFAULT NULL,
  `OrderFiscalWeek` int(11) DEFAULT NULL,
  `DayOfWeek` int(11) DEFAULT NULL,
  PRIMARY KEY (`Order_Date_SK`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
/*!40101 SET character_set_client = @saved_cs_client */;
```

Product DIM Creation and Constraints

```
DROP TABLE IF EXISTS `product_dim`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `product_dim` (
  `Product_SK` int(11) NOT NULL,
  `ProductID` int(11) DEFAULT NULL,
  `ProductName` varchar(50) DEFAULT NULL,
  `Price1` double DEFAULT NULL,
  `Price2` double DEFAULT NULL,
  `UnitCost` double DEFAULT NULL,
  `ProductTypeID` int(11) DEFAULT NULL,
  `SupplierID` int(11) DEFAULT NULL,
  `SupplierName` varchar(100) DEFAULT NULL,
  `SupplierAddr1` varchar(100) DEFAULT NULL,
  `SupplierAddr2` varchar(100) DEFAULT NULL,
  `SupplierCity` varchar(100) DEFAULT NULL,
  `SupplierState` varchar(100) DEFAULT NULL,
  `SupplierZip` char(5) DEFAULT NULL,
  `DivisionID` int(11) DEFAULT NULL,
  `ProductTypeDescription` varchar(100) DEFAULT NULL,
  `BUID` varchar(20) DEFAULT NULL,
  `BUName` varchar(100) DEFAULT NULL,
  `BUAbbrev` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`Product_SK`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
/*!40101 SET character_set_client = @saved_cs_client */;
```

SaleDate DIM Creation and Constraints

```
DROP TABLE IF EXISTS `saledate_dim`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `saledate_dim` (
  `SalesDate_SK` int(11) NOT NULL,
  `SalesDate` text,
  `SalesYear` int(11) DEFAULT NULL,
  `SalesQuarter` int(11) DEFAULT NULL,
  `SalesMonth` int(11) DEFAULT NULL,
  `SalesWeek` int(11) DEFAULT NULL,
  `SalesDay` int(11) DEFAULT NULL,
  `DayOfWeek` int(11) DEFAULT NULL,
  `SalesFiscalYear` int(11) DEFAULT NULL,
  `SalesFiscalQuarter` int(11) DEFAULT NULL,
  `SalesFiscalMonth` int(11) DEFAULT NULL,

  `SalesFiscalWeek` int(11) DEFAULT NULL,

  PRIMARY KEY (`SalesDate_SK`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

/*!40101 SET character_set_client = @saved_cs_client */;
```

<u>Sales Fact Creation and Constraints</u>

```
DROP TABLE IF EXISTS `sales_fact`;

/*!40101 SET @saved_cs_client     = @@character_set_client */;

/*!40101 SET character_set_client = utf8 */;

CREATE TABLE `sales_fact` (

  `Order_Date_SK` int(11) NOT NULL,

  `Sales_Date_SK` int(11) NOT NULL,

  `Product_SK` int(11) NOT NULL,

  `Payment_Order_Shipping_JunkSK` int(11) NOT NULL,

  `Customer_SK` int(11) NOT NULL,

  `Amount` decimal(10,2) DEFAULT NULL,

  `Quantity` int(11) DEFAULT NULL,

  `ShipCost` decimal(10,2) DEFAULT NULL,

  `Discounted` int(11) DEFAULT NULL,

  `InvoiceID_DD` int(11) DEFAULT NULL,
  `Profit` DECIMAL(10,2) DEFAULT NULL,
  `Number_of_days_to_ship` INT(11) DEFAULT NULL,
  PRIMARY KEY
(`Order_Date_SK`,`Sales_Date_SK`,`Product_SK`,`Payment_Order_Shipping_JunkSK`,`Customer_SK`),
  KEY `Customer_SK_idx` (`Customer_SK`),
  KEY `product_sk_idx` (`Product_SK`),
  KEY `saledate_SK_idx` (`Sales_Date_SK`),
  KEY `junk_SK_idx` (`Payment_Order_Shipping_JunkSK`),
  KEY `order_SK_idx` (`Order_Date_SK`),
  CONSTRAINT `Customer_SK` FOREIGN KEY (`Customer_SK`) REFERENCES `customer_dim` (`Customer_SK`) ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `Product_SK` FOREIGN KEY (`Product_SK`) REFERENCES `product_dim` (`Product_SK`) ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `junk_sk` FOREIGN KEY (`Payment_Order_Shipping_JunkSK`) REFERENCES `shipping_payment_order_junk_dim` (`Payment_Order_Shipping_JunkSK`) ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `orderDate_sk` FOREIGN KEY (`Order_Date_SK`) REFERENCES `orderdate_dim` (`Order_Date_SK`) ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `order_sk` FOREIGN KEY (`Order_Date_SK`) REFERENCES
`orderdate_dim` (`Order_Date_SK`) ON DELETE NO ACTION ON UPDATE NO
ACTION,
  CONSTRAINT `sale_SK` FOREIGN KEY (`Sales_Date_SK`) REFERENCES
`saledate_dim` (`SalesDate_SK`) ON DELETE NO ACTION ON UPDATE NO
ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
/*!40101 SET character_set_client = @saved_cs_client */;
```

Shipping_Payment_Order_Junk DIM Creation and Constraints

```
DROP TABLE IF EXISTS `shipping_payment_order_junk_dim`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `shipping_payment_order_junk_dim` (
  `Payment_Order_Shipping_JunkSK` int(11) NOT NULL,
  `ShippingMethod` varchar(45) DEFAULT NULL,
  `PaymentMethod` varchar(45) DEFAULT NULL,
  `OrderMethod` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`Payment_Order_Shipping_JunkSK`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
/*!40101 SET character_set_client = @saved_cs_client */;
```

VI. Data Staging Activities - ETL

1. Data Cleansing

DM Table	**Attribute**	**Problem**	**Resolution Strategy** (attach code)
Customer:			
PEC_Customer	Custtype	There are double and single commas in cust type	The commas are removed by using “Replace in string” from Pentaho.
PEC_Customer	Custtype	For Customer ID 33, custtype is wrongly spelled as “COMERCIAL”	The custtype is changed by using a value mapper in pentaho from “COMERCIAL” to “COMMERCIAL”
PEC_Customer	Custtype	Custtype is in all CAPS. To maintain consistency with PEC_CustomerType file Custype is made all lowercase.	A value mapper is used in pentaho to convert uppercase custtype column to lowercase.

PEC_Customer TPCW_Customer	address	TPCE_customer have two address fields as addr1 and addr2. To maintain consistency address field is splitted into two field, addr1 being the primary address and add2 being the optional	Add constant from pentaho is used to create a new address field and rename them to addr1 and addr2. Addr1 being the main address and addr2 being optional.
PEC_Customer	address	Address field has P.O. Box 825 for CustID 4 which needs to be split	P.O. box is manually moved to addr2 column for this instance
PEC_Customer	address	Address field has inconsistent notations such as Rd. , Av., Ave , St. , dr.	The short forms are converted to full forms to maintain consistency such as Rd. is converted to road, Av. is converted to avenue, St. is converted to street, Dr. is converted to drive
TPCW_Customer	Address	Address field had suite number and department number concatenated in it.	The Suite # and Dept # are splitted so that these are in addr2 field manually

PEC_Customer, TPCE_Customer	address(addr1)	There are extra dots and commas in address Example: 6836 At, Rd. 1792 Squash. Drive	The extra dots and commas are removed in both the files by using replace in string from pentaho.
PEC_Customer TPCE_Customer TPCW_Customer	Zipcode	CustID: 40 has 4- and 3-digit zip code 7066, 778 for cust ID 15 in TPCE_Customer etc.	Zip code is made 5-digit string by appending appropriate number of zeros in front of a zipcode length less than 5 by using java script in pentaho
PEC_Customer, TPCE_Customer, TPCW_Customer	name	The name of the customer has inconsistent abbreviations such as Corp, Corp., Corporation , Company, Co., Inc., Inc	All abbreviations are made consistent by changing Corporation to Corp. , Company to Co. , Inc to Inc. , Incorporated to Inc. etc using replace in string in pentaho
PEC_CustomerType	Typename	The type name when read through pentaho had quotes(“) around it	The quotes are removed using “Replace in string” from Pentaho.

TPCE_CustomerType	typename	Typename S has name as “State_Local Gov”	Name is changed to State/Local Gov to maintain consistency
TPCW_Customer	custtype	The custtype has extra commas at the end	Extra commas are removed using replace in string in pentaho
TPCW_Customer	custtype	The cust type are abbreviated as State , Comm, Commm, Edu and Govt	The custtype are mapped using value mapper as follows in pentaho: Comm, Commm : Commercial Edu : Education Govt : US Govt State : State/Local Govt
TPCW_Customer	State	The state names are are not abbreviated as other tables.	The state names are abbreviated using value mapper. For example, Wyoming to WY, Texas to TX etc.

TPCW_Customer, TPCE_Customer,PEC Customer	All column names	All the column names are made consistent in all the files	All the column names are made consistent in all the files . Some manually some using pentaho
TPCW_CustomerType, TPCE_CustomerType,PEC CustomerType	All column names	All the column names are made consistent in all the files	All the column names are made consistent in all the files . Some manually some using pentaho
Product:			
TPCW_Product	All Attributes	No Header	Added Column Names while Taking the input file in Pentaho.
TPCW_Product, TPCE_Product,	All Attributes	All the attributes were enclosed in double quotes(“).	Used the transformation Replace in strings in

PEC_Product, TPCW_ProductType TPCE_ProductType PEC_ProductType			Pentaho to remove the double quotes
TPCW_Product	ProductName	Has Duplicates at the respective ProductID's: 1. 200 with ProductID 90 (Curiouser Cleaning Supplies) 2. 106 with 70 (Escape Manufacturing Equipment) 3. 102 with 78 (Measured Photo Chemicals) 4. 101 with 17(Optima Cleaning Supplies)	Removed the Duplicate rows at Following ProductID's: 200,101,102,106.

TPCW_Product, TPCE_Product,	ProductName	Product Name Equip was incomplete for some of the ProductNames.	Changed Equip to Equipment in Excel.
TPCW_Product	SupplierAddress	Incomplete spelling, short forms used and incorrect entries in the address as follows, Ave, st, careless,6027,3237	Used Pentaho transformation to make following changes using replace in string: Ave-Avenue St-Street Careless-Carelessly 6027-6037 3237-3727
TPCW_Product	SupplierState, SupplierCity	SupplierState and SupplierCity were a single field.	Split the SupplierState and SupplierCity using split transformation in Pentaho.
TPCW_Product	SupplierAddress	The Address was a single field	Split the Address in Addr1 and Addr2.

TPCW_Product	SupplierState	Some State Abbreviations were inconsistent (Both letters not capital). States: Fl,Va,Wa,Pa,Mn,Mi,Ky.	Changed the States Fl,Va,Wa,Pa,Mn,Mi,Ky to capital letters FL, VA, WA, PA, MN, MI, KY to make them consistent in Excel.
TPCW_Product	SupplierName	Following Names different in different tables(With reference to customer tables): Corporation, Inc	Changed Corporation to corp. And Inc to Inc. in Excel for consistency in names in the table.
TPCW_Product	SupplierCity	One of the Cities 'Tallahassee' was misspelled as 'Talahassee' at ProductID: 97.	Corrected the spelling mistake to 'Tallahassee' in Excel.
TPCE_Product	ProductTypeID	Have Leading zeros before the ProductTypeID	Used Modified Java script in Pentaho to remove the leading zeros.
PEC_Product	All Attributes except	Enclosed in double quotes	Removed double quotes using replace in string transformation in Pentaho

	Price1 and Price 2		
PEC_Product	ProductTypeID	ProductTypeID 33 out of range.All other ProductTypeID lie between (1,15)	Changed ProductTypeID 33 to 3 in Excel.
PEC_Product	UnitCost	Missing Values for UnitCost in PEC_Product.	Calculated the missing UnitCost from PEC Manufacturing cost and PEC Invoice.
TPCW_ProductType,PEC_ProductType	TypeDescription	Column name was in capital letters TYPEDESCRIPTION	Changed the column Name from TYPEDESCRIPTION to TypeDescription to follow standard nomenclature.
TPCW_ProductType,PEC_ProductType	ProductTypeID	The column of ProductTypeID was PRODTYPEID	Changed the column Name from PRODTYPEID to ProductTypeID to make it consistent with the name in the Product table.

TPCW_ProductType,PEC_Product Type	ProductTypeID	Had leading Zeros	Removed all leading zeros in the ProductTypeID using Modified java script in Java.
TPCW_BU, PEC_BU	BUName,BUAbbrev	Attributes were enclosed in double quotes.	Used the transformation Replace in strings in Pentaho to remove the double quotes
TPCW_BU. PEC_BU	BUID	Leading Space before each BUID	Removed the space using trim both in pentaho transformation.
TPCW_BU, TPCE_BU PEC_BU	BUName,BUAbbrev	Columns Names did not specify BU	Added BU to the column names Name and ABBREV and changed it to BUName and BUAbbrev in excel.
TPCW_BU, TPCE_BU, PEC_BU	BUAbbrev	The Abbreviation for BUName Miscellaneous was missing.	Replaced the null value for BUAbbrev as Misc using if field Null transformation in Pentaho.
Invoice:			

PEC_Invoice	salesDate	For Invoice ID 72, the date was not in the right format - 200805-16	Changed the date in Excel to 05/16/2008
PEC_Invoice	shipMethod	Spelling mistake in shipping method values like A. In InvoiceID 52778, the shipMethod is ‘aiir’. B. In InvoiceID, 3432, the shipMethod is ‘trran. C. In InvoiceID 37461, the shipMethod is ‘trick’. D. In InvoiceID 43751, the shipMethod is ‘tuck’. E. In InvoiceID 38432, the shipMethod is ‘trrain’.	Updated the spellings in Excel.

PEC_Invoice	amt	The amounts were wrong in comparison to the quantity purchased	Updated the amounts as per discounted flag and Price1 and Price2 attributes in Pentaho using JavaScript. 1. Discounted 0, then considered Price1. 2. Discounted 1, then considered Price2.
PEC_Invoice	prodid	The columns with InvoiceID 12485 and 0000025563"11 had weird column values	Shifted columns in Excel made the values of SalesDate, OrderDate correct.
TPCW_Invoice	invoice	The column values for invoiceID 26511 had invalid values like custID, salesDate, amt, qty	Deleted the record in Excel
TPCW_Invoice	custID	Negative custIDs -14 and -8 for invoice IDs 21923 and 23492	Changed custIDs from negative to positive in Excel

TPCW_Invoice	prodID	Some prodIDs were not valid as per product’s file id like 101, 102 and 399	Replaced 101 with 17, 102 with 78 and 399 with 99 in Pentaho using JavaScript
TPCW_Invoice	salesDate	No uniformity in the date formats. Formats found: 31-12-08, 6/7/2007 etc.	Changed the date format to MM/DD/YYYY in Pentaho
TPCW_Invoice	amt	The amounts were wrong in comparison to the quantity purchased	Updated the amounts as per discounted flag and Price1 and Price2 attributes in Pentaho using JavaScript.
TPCW_Invoice	prodID	Invalid prodID 41 in invoice 14710 as per Product table	Changed it to 40 in Excel
TPCW_Invoice	invoiceID	invoice 3032 had the following problems: 1. Qty is missing. 2. salesDate has value 20-08-05372	Shifted the record in Excel hence the values became salesDate 20-08-05 Amt 372

			Qty 52
TPCW_Invoice	discounted	Discounted value missing in invoice 45461	Discounted changed to 1 in Excel

2. Data Transformation

DM Table	**Image Creation Process** (attach code)
Customer_Dim	1. Extract all customer and customer_type files for three divisions. 2. Clean them so as to take out extra commas, full stops, and spelling mistakes. Make customer name consistent in terms of abbreviations such as Company vs Co. . We have kept the abbreviations. 3. Map the customer type of all division so that they are consistent 4. Merge Customer and Cusomer_Type for all the three divisions. 5. Add ADDR1 column to TPCW and PEC. Make addr1 as main address and addr2 as optional address(Dept No, Suite No, P.O. BOX) 6. Map states to its abbreviation in TPCW 7. Replace the abbreviations such as Ave, Rd, Dr in address 2 with complete name 8. Add DivisionID (1 for TPCE, 2 for TPCW and 3 for PEC) 9. Merge all the three division tables 10. Add surrogate keys and send it to output file

	There are same customers doing business with different departments. We have not removed those customers so that they can be queried to see division wise as well as overall business by just changing the group by statement. Transformations can be found in Customer_transformation.ktr
Product_Dim	1. Extract the pre-cleaned TPCW, TPCE Product and TPCE_Supplier files. Split the Address field into Addr1 and Addr2 respectively. Add Supplier details to the TPCE file to make it consistent with TPCW file 2. Add DivisionID field to both the tables such that TPCE:DivisionID = 1 TPCW:DivisionID = 2 3. Create TPCW_TPCE_Product file by merging the two files. 4. Extract TPCW_TPCE_Product, TPCW_BU, and TPCW_ProductType files,clean them as explained in Data cleansing step and merge all the files. 5. Extract PEC_Product, PEC_BU,PEC_ManufacturingCost ,PEC_invoice and calculated unit cost as Total Quantity/Total Cost from the PEC_ManufacturingCost and PEC_Invoice tables. Merge this calculated Unit cost in the PEC_product where UnitCost is null.

	6. Merge PEC_Product, PEC_Product_Type and PEC_BU. 7. Add Supplier details to the PEC table to make the number of fields consistent with TPCW_TPCE table. 8. Add DivisionID for PEC:DivisionID = 3 10. Append the tables TPCW_TPCE and PEC. 11. Add Surrogate Key. 12.Rename,Rearrange/Reorder all the fields to make them consistent and change the Data Types to respective formats for all fields. 13. Export the Output file **Product.csv** Transformations can be found in FinalTransformation_Product.ktr
Invoice_DD	1. Extracted the pre-cleaned TPCW Invoice, PEC Invoice, TPCE Invoice and TPCE Invoice Detail files. 2. Removed unwanted attributes like Shipping Method, Order Date, SalesDate, Payment Method etc. for all the three divisions. 3. Sorted all the files based on InvoiceID. 4. Merged TPCE Invoice and TPCE Invoice Details. 5. Add DivisionID field to both the tables such that ● TPCE:DivisionID = 1 ● TPCW:DivisionID = 2 ● PEC:DivisionID = 3 6. Added ShipCost column with value = 0.0 in TPCE and TPCW.

	7. Changed Amount by calculating it in Javascript using Price1 and Price 2 values from Product Table and Discounted value from Invoice. 8. Appended the tables TPCW_TPCE and PEC. 9. Add Surrogate Key. 10. Renamed,Rearranged/Reordered all the fields to make them consistent and changed the Data Types to respective formats for all fields. 11. Export the Output file in **Cleaned_Invoice.csv** Transformations can be found CleanedInvoice.ktr
OrderDate_Dim	1. Extracted the pre-cleaned PEC Invoice. 2. Removed unwanted attributes like Shipping Method, Sales Date, amt, qty, Payment Method etc.. 3. Used Calculator in Pentaho to calculate values of Order Year, Order Month, Order Quarter, Order Day and DayOfWeek from order date. 4. Added Javascript code for calculating Fiscal Year, Fiscal Quarter, Fiscal Month and Fiscal Week based on order date. 5. Add Surrogate Key. 6. Added null rows for TPCE and TCPW as they do not contain Order Date with OrderDate_SK of 9000 and 9001 respectively.. 7. Add DivisionID field to both the tables such that

	● TPCE:DivisionID = 1 ● TPCW:DivisionID = 2 ● PEC:DivisionID = 3 8. Appended the tables TPCW_TPCE and PEC. 9. Renamed,Rearranged/Reordered all the fields to make them consistent and changed the Data Types to respective formats for all fields. 10. Export the Output file in **OrderDate.csv** Transformations can be found in CleanedOrderDate.ktr
SalesDate_Dim	1. Extracted the pre-cleaned TPCW Invoice, PEC Invoice, TPCE Invoice and TPCE Invoice Detail files. 2. Removed unwanted attributes like Shipping Method, Order Date, amt, qty, Payment Method etc. for all the three divisions. 3. Sorted all the files based on InvoiceID. 4. For TPCW, fixed the date format by using the following: A. Replace in String to replace ‘-’ with ‘/’. B. Split salesDate in Month value, Year Value and DateValue. C. Concatenated the Month value, Year value and Date value in JavaScript so that all the dates are valid and months and dates are not greater than 12 and 31 respectively. D. Changed string order date to date in MM//DD/YY format.

	E. Used Calculator to change the date format to MM/DD/YYYY. 5. Merged TPCE Invoice and TPCE Invoice Details. 6. Add DivisionID field to both the tables such that ● TPCE:DivisionID = 1 ● TPCW:DivisionID = 2 ● PEC:DivisionID = 3 7. Used Calculator in Pentaho to calculate values of Sales Year, Sales Month, Sales Quarter, Sales Day and DayOfWeek from Sales date. 8. Added Javascript code for calculating Fiscal Year, Fiscal Quarter, Fiscal Month and Fiscal Week based on order date. 9. Appended the tables TPCW_TPCE and PEC. 10. Add Surrogate Key. 11. Renamed,Rearranged/Reordered all the fields to make them consistent and changed the Data Types to respective formats for all fields. 12. Export the Output file in **SalesDate.csv** Transformations can be found in CleanedSalesDate.ktr
Junk_Dim	1. Created a table with 36 rows with all the possible combinations of Shipping Method, Order Method and Payment Method due to low cardinality.

Sales_fact	1. Created a Junk_Prep table. a. Extracted the pre-cleaned PEC Invoice. b. Removed unwanted attributes like Order Date, amt, qty, Sales Date, Ship Cost etc.. c. Removed Duplicates d. Merged the file created in step a to assign SKs according to the 36 possible combinations of Shipping Method, Order Method and Payment Method (Transformations can be found in CleanedMiscJunkDimension.ktr) 2. Extracted the cleaned Product, Customer, Invoice, Sales Date, Order Date, Junk and Junk Prep CSV files. 3. Merged Invoice and Order Date based on InvoiceID, CustomerID and ProductID. 4. Merged the resultant table with Sales Date based on InvoiceID, CustomerID, ProductID. 5. Merged Junk_Prep and Junk Table in order to assign the correct Junk_Dim SKs to the Invoices based on combination of Shipping Method, Payment Method and Order Method. 6. Merged the resultant table in step 3 with step 4 table based on InvoiceID

	7. Merged the resultant table with Product table based on ProductID and Division ID. 8. Finally merged the resultant table with Customer table based on Customer ID and Division ID. 9. Renamed,Rearranged/Reordered all the fields to make them consistent and changed the Data Types to respective formats for all fields. 10. Export the Output file **Sales_Fact.csv** Transformations can be found in Fact_Table.ktr

3. Table Population

DM Table	**Table Population Process** (attach code)
Customer_Dim	-- Load Customer Dimension LOAD DATA LOCAL INFILE '/Users/varunchaudhary/Documents/Lab3_185/Lab3_DataFiles_185/Customer/Output/Customer_cleaed_for_sql.csv' INTO TABLE customer_dim FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"' LINES TERMINATED BY '\r\n' IGNORE 1 LINES;

SaleDate_Dim	-- Load SalesDate Dimension LOAD DATA LOCAL INFILE '/Users/varunchaudhary/Documents/Lab3_185/Lab3_DataFiles_185/ Sales_Date/Output/sale_date_sql.csv' INTO TABLE saledate_dim FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"' LINES TERMINATED BY '\n' IGNORE 1 LINES (SalesDate_SK, @SalesDate, SalesYear, SalesQuarter, SalesMonth, SalesWeek, SalesDay, DayOfWeek, SalesFiscalYear, SalesFiscalQuarter,SalesFiscalMonth, SalesFiscalWeek) set salesDate = STR_TO_DATE(@salesDate, '%m/%d/%YY');
Orderdate_Dim	-- Load OrderDate Dimension LOAD DATA LOCAL INFILE '/Users/varunchaudhary/Documents/Lab3_185/Lab3_DataFiles_185/ Order_Date/Output/output_for_sql_order_Date.csv' INTO TABLE orderdate_dim FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"' LINES TERMINATED BY '\n'

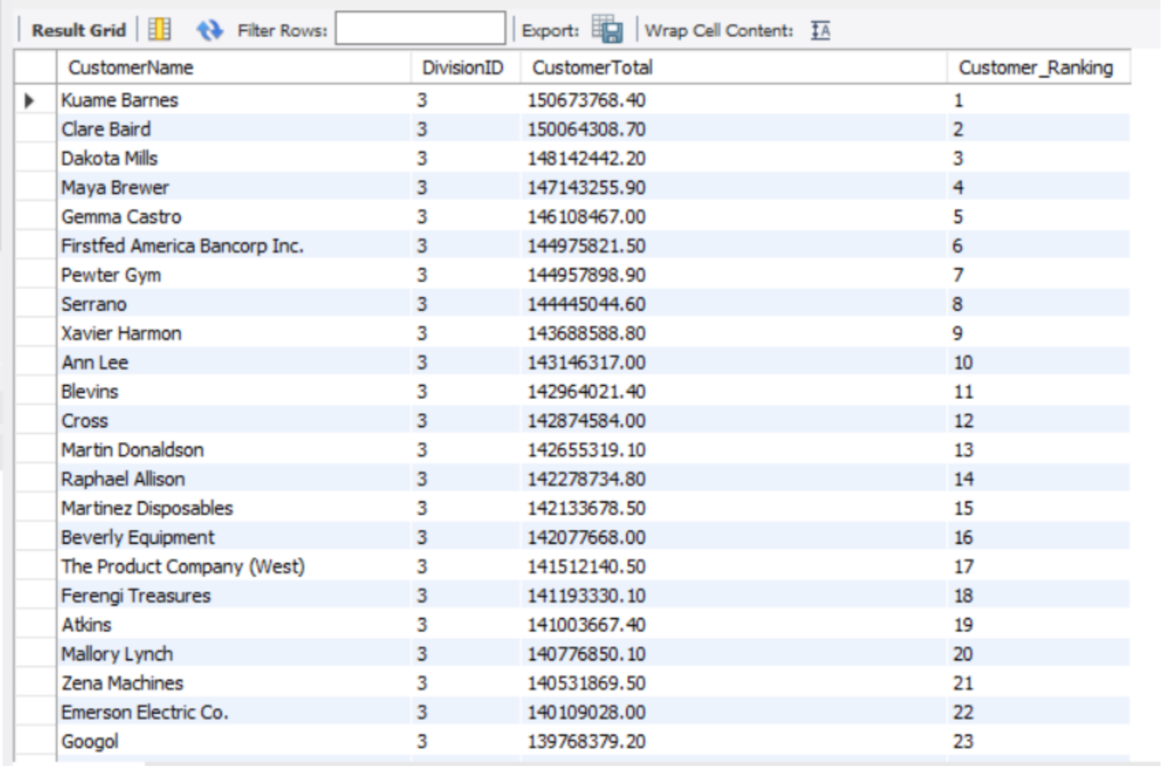
	IGNORE 1 LINES (Order_Date_SK, @orderDate, OrderYear, OrderQuarter, OrderMonth, OrderWeek, OrderDay, OrderFiscalYear, OrderFiscalQuarter,OrderFiscalMonth, OrderFiscalWeek,DayOfWeek) set orderDate = STR_TO_DATE(@orderDate, '%m/%d/%YY');
Product_Dim	-- Load Product Dimension LOAD DATA LOCAL INFILE '/Users/varunchaudhary/Documents/Lab3_185/Lab3_DataFiles_185/ Product/Output/Product_Cleaned_for_sql.csv' INTO TABLE product_dim FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"' LINES TERMINATED BY '\r\n' IGNORE 1 LINES;
Shipping_Payment_Order_Junk_Dim	-- Load Junk Dimension LOAD DATA LOCAL INFILE '/Users/varunchaudhary/Documents/Lab3_185/Lab3_DataFiles_185/ junk/input/Shipping_Payment_Order_Junk_Dimension.csv' INTO TABLE shipping_payment_order_dimension FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'

	LINES TERMINATED BY '\r\n' IGNORE 1 LINES;
Sales_fact	-- Load Fact Table LOAD DATA LOCAL INFILE '/Users/varunchaudhary/Documents/Lab3_185/Lab3_DataFiles_185/fact/output/fact_output.csv' INTO TABLE sales_fact FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"' LINES TERMINATED BY '\r\n' IGNORE 1 LINES;
NOTE:	The customer files when opened in excel shows some zip coded as three or four digits because it considers the column as general not text. The fixed zip codes with zero prepended can be seen when while is opened as text file. The Customer ID, Product ID columns are removed manually from fact table before uploading it in data mart.

VII. End User Applications

1. Queries

User Question/Reporting Need
#Query 1: Rank Customer on the basis of total sales
SQL Code
select a.CustomerName, a.DivisionID, a.CustomerTotal, count(b.CustomerTotal) Customer_Ranking from (select Customer_SK, CustomerName, DivisionID, sum(Amount) CustomerTotal from sales_fact join customer_dim using (Customer_SK) group by CustomerName, DivisionID) a, (select Customer_SK, CustomerName, DivisionID, sum(Amount) CustomerTotal from sales_fact join customer_dim using (Customer_SK) group by CustomerName, DivisionID) b where a.CustomerTotal <= b.CustomerTotal group by a.Customer_SK order by 4;
Supporting Index(es)
Customer_SK from Customer Dimension
Output



CustomerName	DivisionID	CustomerTotal	Customer_Ranking
Kuame Barnes	3	150673768.40	1
Clare Baird	3	150064308.70	2
Dakota Mills	3	148142442.20	3
Maya Brewer	3	147143255.90	4
Gemma Castro	3	146108467.00	5
Firstfed America Bancorp Inc.	3	144975821.50	6
Pewter Gym	3	144957898.90	7
Serrano	3	144445044.60	8
Xavier Harmon	3	143688588.80	9
Ann Lee	3	143146317.00	10
Blevins	3	142964021.40	11
Cross	3	142874584.00	12
Martin Donaldson	3	142655319.10	13
Raphael Allison	3	142278734.80	14
Martinez Disposables	3	142133678.50	15
Beverly Equipment	3	142077668.00	16
The Product Company (West)	3	141512140.50	17
Ferengi Treasures	3	141193330.10	18
Atkins	3	141003667.40	19
Mallory Lynch	3	140776850.10	20
Zena Machines	3	140531869.50	21
Emerson Electric Co.	3	140109028.00	22
Googol	3	139768379.20	23

User Question/Reporting Need

#Query 2: Percentage of sales order by Payment Method

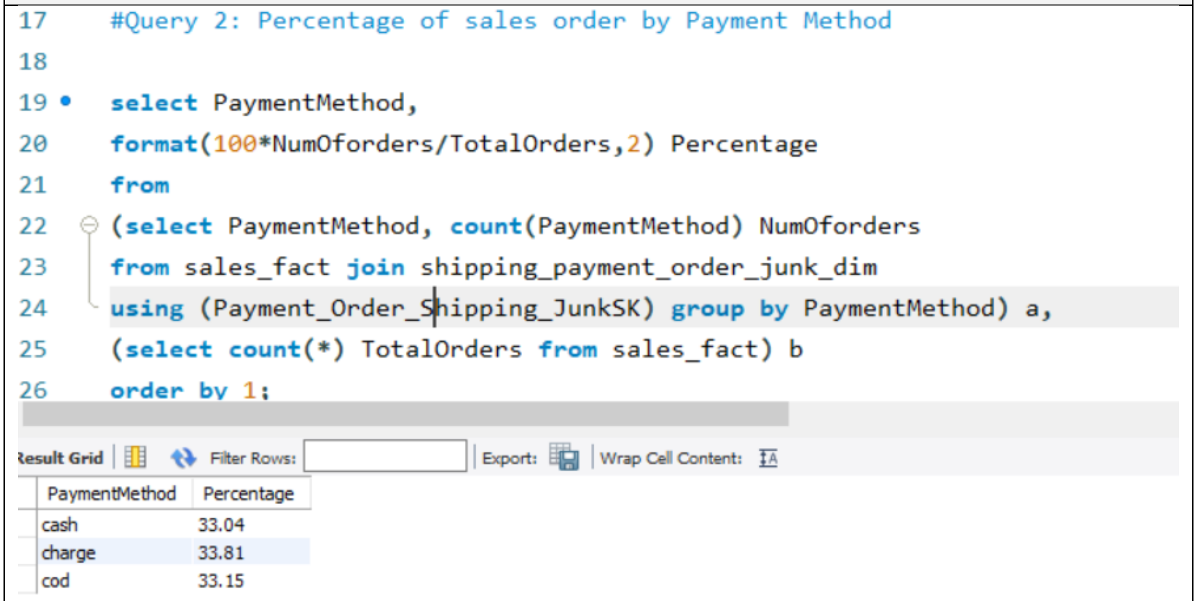
SQL Code

```
select PaymentMethod,

format(100*NumOforders/TotalOrders,2) Percentage

from

(select PaymentMethod, count(PaymentMethod) NumOforders

from sales_fact join shipping_payment_order_junk_dim

using (Payment_Order_Shipping_JunkSK) group by PaymentMethod) a,

(select count(*) TotalOrders from sales_fact) b

order by 1;
```

Supporting Index(es)

Payment_order_shipping_junk_SK from Junk Dimension

Output



```
#Query 2: Percentage of sales order by Payment Method

select PaymentMethod,
format(100*NumOforders/TotalOrders,2) Percentage
from
(select PaymentMethod, count(PaymentMethod) NumOforders
from sales_fact join shipping_payment_order_junk_dim
using (Payment_Order_Shipping_JunkSK) group by PaymentMethod) a,
(select count(*) TotalOrders from sales_fact) b
order by 1;
```

PaymentMethod	Percentage
cash	33.04
charge	33.81
cod	33.15

User Question/Reporting Need

#Query 3: The most frequent Order Method in PEC.

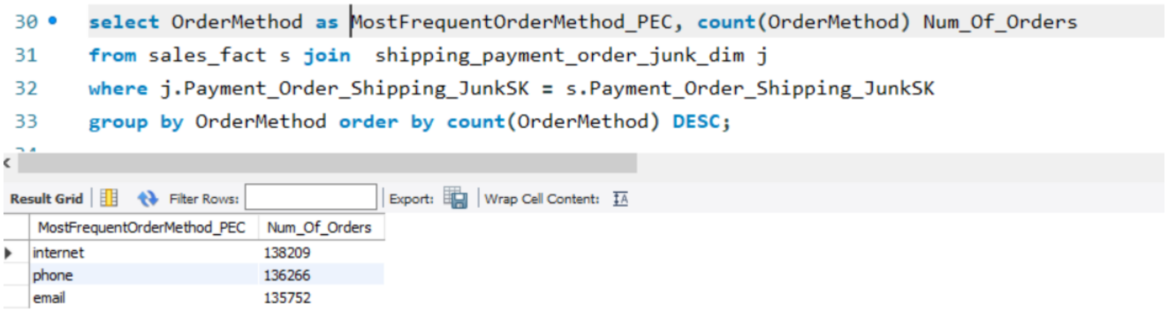
SQL Code

```
select OrderMethod as MostFrequentOrderMethod_PEC, count(OrderMethod)

Num_Of_Orders

from sales_fact s join  shipping_payment_order_junk_dim j

where j.Payment_Order_Shipping_JunkSK = s.Payment_Order_Shipping_JunkSK

group by OrderMethod order by count(OrderMethod) DESC;
```

Supporting Index(es)

Payment_order_shipping_junk_SK from Junk Dimension

Outputs



```sql
select OrderMethod as MostFrequentOrderMethod_PEC, count(OrderMethod) Num_Of_Orders
from sales_fact s join  shipping_payment_order_junk_dim j
where j.Payment_Order_Shipping_JunkSK = s.Payment_Order_Shipping_JunkSK
group by OrderMethod order by count(OrderMethod) DESC;
```

```
MostFrequentOrderMethod_PEC   Num_Of_Orders
internet                      138209
phone                         136266
email                         135752
```

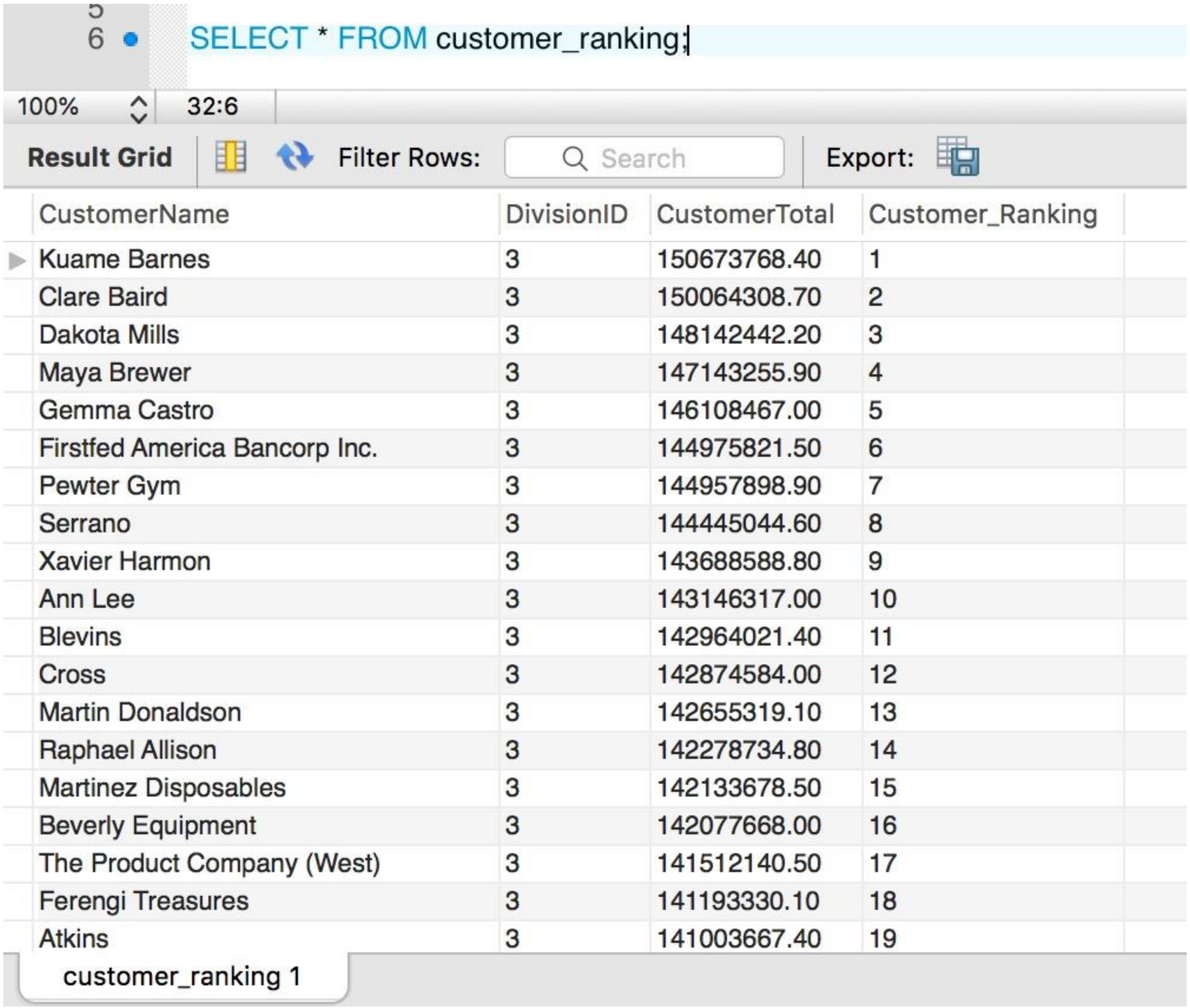
2. A View

View Customer_Ranking will store the view of the records which stores the ranking of Customers on the basis of total sales.

View Creation:

```
create view Customer_Ranking as

select a.CustomerName, a.DivisionID, a.CustomerTotal, count(b.CustomerTotal)

Customer_Ranking

from

(select Customer_SK, CustomerName, DivisionID, sum(Amount) CustomerTotal

from sales_fact join customer_dim using (Customer_SK) group by CustomerName,

DivisionID) a,

(select Customer_SK, CustomerName, DivisionID, sum(Amount) CustomerTotal

from sales_fact join customer_dim using (Customer_SK) group by CustomerName,

DivisionID) b

where a.CustomerTotal <= b.CustomerTotal

group by a.Customer_SK

order by 4;
```

Output:



```
SELECT * FROM customer_ranking;
```

CustomerName	DivisionID	CustomerTotal	Customer_Ranking
Kuame Barnes	3	150673768.40	1
Clare Baird	3	150064308.70	2
Dakota Mills	3	148142442.20	3
Maya Brewer	3	147143255.90	4
Gemma Castro	3	146108467.00	5
Firstfed America Bancorp Inc.	3	144975821.50	6
Pewter Gym	3	144957898.90	7
Serrano	3	144445044.60	8
Xavier Harmon	3	143688588.80	9
Ann Lee	3	143146317.00	10
Blevins	3	142964021.40	11
Cross	3	142874584.00	12
Martin Donaldson	3	142655319.10	13
Raphael Allison	3	142278734.80	14
Martinez Disposables	3	142133678.50	15
Beverly Equipment	3	142077668.00	16
The Product Company (West)	3	141512140.50	17
Ferengi Treasures	3	141193330.10	18
Atkins	3	141003667.40	19

3. Aggregated Mata Marts

A. Lost Dimension

This fact is built using Sales Date Dimension. Other dimensions such as Product, Order Date, Junk, Customer are being lost. The dimension formed contains the total sales amount for a particular date.

Population of DataMart

```
create table sales_by_date_fact

select SalesDate_SK, SalesDate, sum(Amount) as TotalAmount, Sum(Quantity) as

TotalQuantity

from sales_fact f join saledate_dim o on f.Sales_Date_SK  = o.SalesDate_SK

group by SalesDate;
```

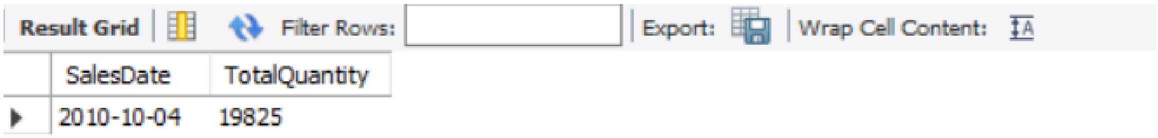
Summary Queries

Use Case 1: If the user wants to know how many products were sold on a particular date.

Sample Query 1: Total Number of products sold on a particular date eg: 2010-10-04

```
select SalesDate, TotalQuantity from sales_by_date_fact where SalesDate = '2010-10-

04';
```

Output:



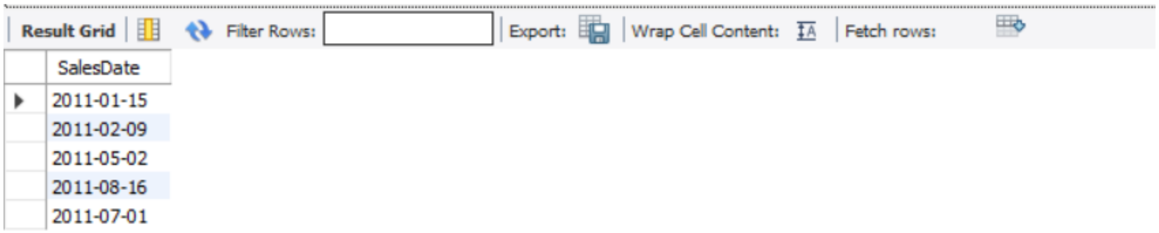
```
SalesDate     TotalQuantity
2010-10-04    19825
```

Use Case 2: If the user wants to know the dates of an year on which maximum products were sold and then the user can analyze why one those particular dates highest sales were made.

Sample Query 2: Top 5 dates in 2011 on which maximum products were sold

```
select SalesDate from sales_by_date_fact

where SalesDate > '2010-12-31' and SalesDate < '2012-01-01'

order by TotalQuantity limit 5;
```

Output:



SalesDate
2011-01-15
2011-02-09
2011-05-02
2011-08-16
2011-07-01

B. Shrunken Dimension

This aggregate fact uses shrunken dimension in order to get quarterly sales. The sales date dimension has been shrunken to Quarter grain level.

Population of DataMart

```
-- create dimension Quarter_Sales_Dim

CREATE TABLE Quarter_Sales_Dim ( New_Sale_SK int NOT NULL

AUTO_INCREMENT, salesQuarter int NOT NULL, salesYear int, PRIMARY KEY

(New_Sale_SK));
```

```
-- load data into shrunken dimension first
insert into Quarter_Sales_Dim (salesQuarter, salesYear) select  SalesQuarter,
SalesYear from saledate_dim group by  SalesYear,SalesQuarter;
-- create fact table for shrunken dimension
create table sales_Quarter_fact as(select
w.Product_SK,w.Payment_Order_Shipping_JunkSK, w.Order_Date_SK
,q.New_Sale_SK, w.TotalQuarterlySales, DivisionID
from (select f.Product_SK, f.Payment_Order_Shipping_JunkSK, f.Order_Date_SK,
s.salesQuarter, s.Salesyear, DivisionID,
sum(f.Amount) as TotalQuarterlySales from sales_fact f
join orderdate_dim o on f.Order_Date_SK = o.Order_Date_SK
join product_dim p on f.Product_SK = p.Product_SK
join saledate_dim s on f.Sales_Date_SK = s.SalesDate_SK
join shipping_payment_order_junk_dim j on f.Payment_Order_Shipping_JunkSK =
j.Payment_Order_Shipping_JunkSK
group by s.salesQuarter, s.Salesyear,p.DivisionID,p.productName) w
join Quarter_Sales_Dim q on w.salesQuarter = q.salesQuarter and w.salesyear =
q.Salesyear);
```

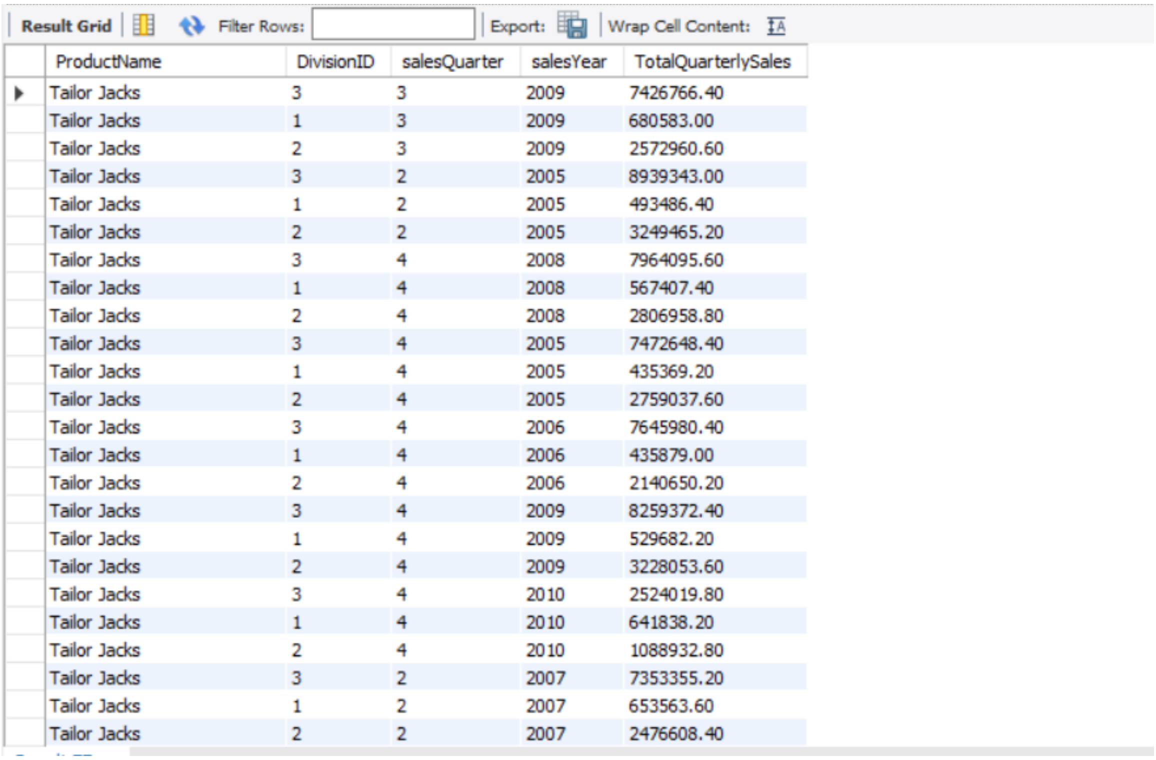
Summary Queries

Use Case 1: If the user wants to know the sale of a product per quarter per year in different divisions. This information can be useful in finding out what is the total sale amount of the product in each division, in which quarter are the most sales made and in which division was the maximum sale made.

Sample Query 1: Total sales of Product 'Tailor Jacks' for different divisions per quarter of a year

```
select distinct p.ProductName, p.DivisionID,s.salesQuarter, s.salesYear,
f.TotalQuarterlySales
from Product_dim p join sales_Quarter_fact f on p.Product_SK = f.Product_SK
join Quarter_Sales_Dim s on s.New_Sale_SK = f.New_Sale_SK where productName =
'Tailor Jacks' ;
```

Output:



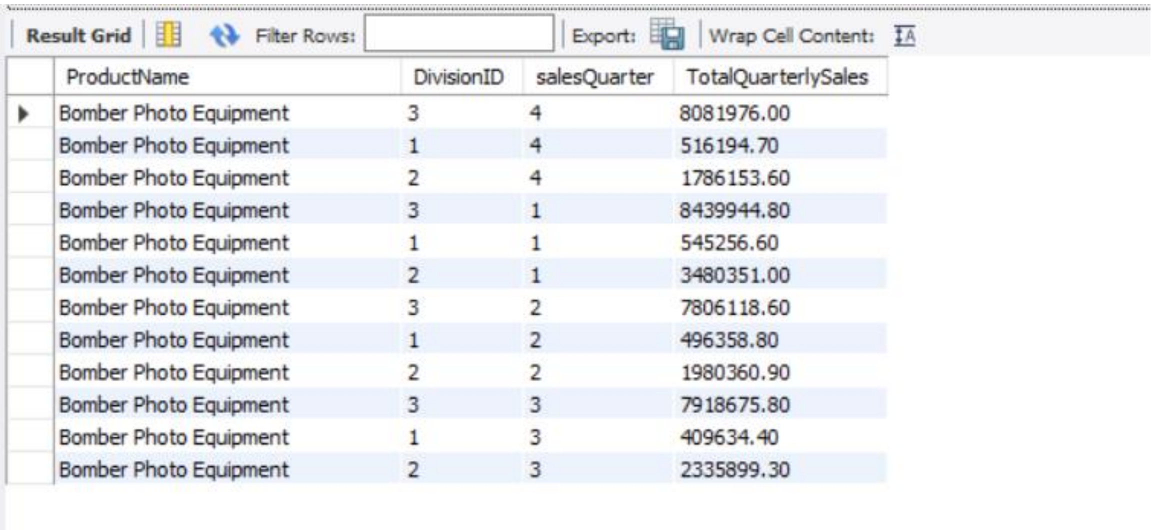
ProductName	DivisionID	salesQuarter	salesYear	TotalQuarterlySales
Tailor Jacks	3	3	2009	7426766.40
Tailor Jacks	1	3	2009	680583.00
Tailor Jacks	2	3	2009	2572960.60
Tailor Jacks	3	2	2005	8939343.00
Tailor Jacks	1	2	2005	493486.40
Tailor Jacks	2	2	2005	3249465.20
Tailor Jacks	3	4	2008	7964095.60
Tailor Jacks	1	4	2008	567407.40
Tailor Jacks	2	4	2008	2806958.80
Tailor Jacks	3	4	2005	7472648.40
Tailor Jacks	1	4	2005	435369.20
Tailor Jacks	2	4	2005	2759037.60
Tailor Jacks	3	4	2006	7645980.40
Tailor Jacks	1	4	2006	435879.00
Tailor Jacks	2	4	2006	2140650.20
Tailor Jacks	3	4	2009	8259372.40
Tailor Jacks	1	4	2009	529682.20
Tailor Jacks	2	4	2009	3228053.60
Tailor Jacks	3	4	2010	2524019.80
Tailor Jacks	1	4	2010	641838.20
Tailor Jacks	2	4	2010	1088932.80
Tailor Jacks	3	2	2007	7353355.20
Tailor Jacks	1	2	2007	653563.60
Tailor Jacks	2	2	2007	2476608.40

Use Case 2: If the user wants to know the sale of a product per quarter for a particular year in different divisions.

Sample Query 2: Total quarterly sales for product 'Bomber Photo Equipment' in 2008

```
select distinct p.ProductName, p.DivisionID,s.salesQuarter, f.TotalQuarterlySales
from Product_dim p join sales_Quarter_fact f on p.Product_SK = f.Product_SK
join Quarter_Sales_Dim s on s.New_Sale_SK = f.New_Sale_SK where productName
= 'Bomber Photo Equipment' and salesYear = '2008';
```

Output:



ProductName	DivisionID	salesQuarter	TotalQuarterlySales
Bomber Photo Equipment	3	4	8081976.00
Bomber Photo Equipment	1	4	516194.70
Bomber Photo Equipment	2	4	1786153.60
Bomber Photo Equipment	3	1	8439944.80
Bomber Photo Equipment	1	1	545256.60
Bomber Photo Equipment	2	1	3480351.00
Bomber Photo Equipment	3	2	7806118.60
Bomber Photo Equipment	1	2	496358.80
Bomber Photo Equipment	2	2	1980360.90
Bomber Photo Equipment	3	3	7918675.80
Bomber Photo Equipment	1	3	409634.40
Bomber Photo Equipment	2	3	2335899.30

C. Collapsed Dimension

In aggregate fact collapsed_fact the Product, Customer Dimension, OrderDate Dimension, aggregated together into one fact table.

<u>Population of DataMart</u>

```
Create table collapsed_fact as (select
c.Customer_SK,p.Product_SK,s.SalesDate_SK,custID,
customerName,SalesDate,salesYear,sum(Amount) as totalAmount, sum(Quantity) as
totalQuantity, ShipCOst, p.DivisionID,ProductID, ProductName
from sales_fact f join product_dim p on p.Product_SK = f.product_SK
join Customer_dim c on c.Customer_SK = f.Customer_SK
join saledate_dim s on s.SalesDate_SK = f.Sales_Date_SK
group by CustID, ProductID, SalesDate);
```

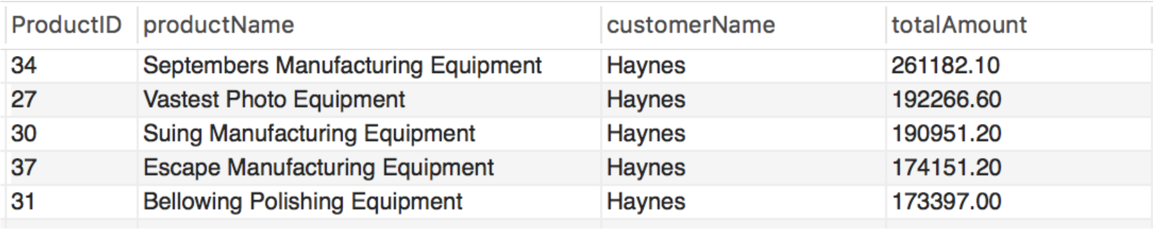
Summary Queries

Use Case 1: If the user wants to know which products a particular customer ordered the most in a particular year. This type of information can be helpful in determining what kind of products the customer is interested in buying.

Sample Query 1: Top Five orders by Customer with name Haynes in 2005

```
select ProductID, productName,customerName, totalAmount from collapsed_fact
where
customerName = 'Haynes' and salesYear = 2009 order by 4 DESC limit 5 ;
```

Output:



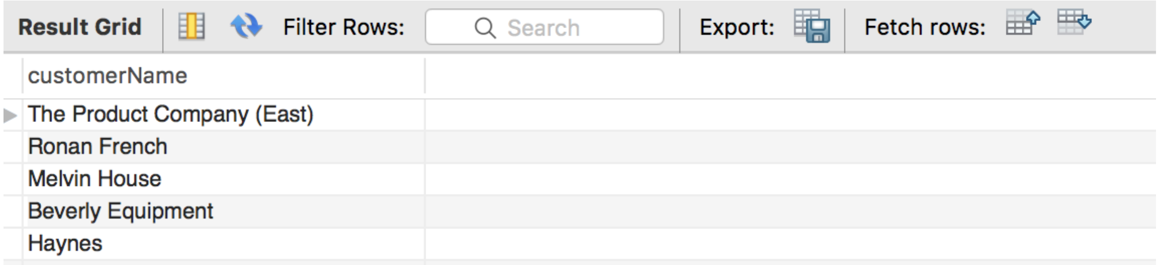
ProductID	productName	customerName	totalAmount
34	Septembers Manufacturing Equipment	Haynes	261182.10
27	Vastest Photo Equipment	Haynes	192266.60
30	Suing Manufacturing Equipment	Haynes	190951.20
37	Escape Manufacturing Equipment	Haynes	174151.20
31	Bellowing Polishing Equipment	Haynes	173397.00

Use Case 2: If the user wants to know which customers bought a **particular product** the most in a particular year.

Sample Query 2: Top 5 customers who orders Product "Automobiles Fillers" in 2008

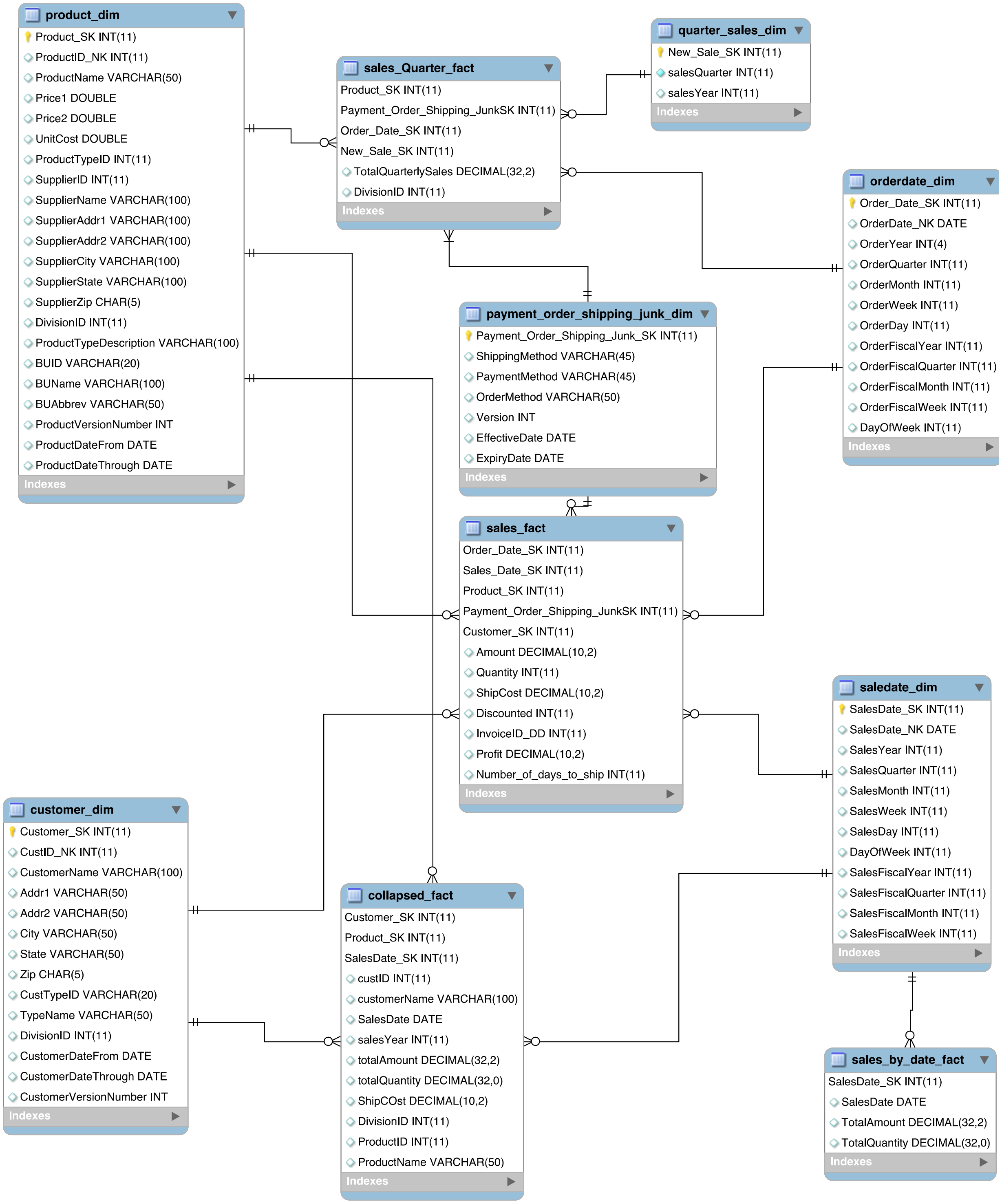
```
select customerName from collapsed_fact where
ProductName = 'Automobiles Fillers' and orderYear = 2008 order by totalAmount
DESC limit 5;
```

Output:



customerName
The Product Company (East)
Ronan French
Melvin House
Beverly Equipment
Haynes

ERD with three aggregated Data Marts:


product_dim
Product_SK INT(11)
ProductID_NK INT(11)
ProductName VARCHAR(50)
Price1 DOUBLE
Price2 DOUBLE
UnitCost DOUBLE
ProductTypeID INT(11)
SupplierID INT(11)
SupplierName VARCHAR(100)
SupplierAddr1 VARCHAR(100)
SupplierAddr2 VARCHAR(100)
SupplierCity VARCHAR(100)
SupplierState VARCHAR(100)
SupplierZip CHAR(5)
DivisionID INT(11)
ProductTypeDescription VARCHAR(100)
BUID VARCHAR(20)
BUName VARCHAR(100)
BUAbbrev VARCHAR(50)
ProductVersionNumber INT
ProductDateFrom DATE
ProductDateThrough DATE
Indexes
sales_Quarter_fact
Product_SK INT(11)
Payment_Order_Shipping_JunkSK INT(11)
Order_Date_SK INT(11)
New_Sale_SK INT(11)
TotalQuarterlySales DECIMAL(32,2)
DivisionID INT(11)
Indexes
quarter_sales_dim
New_Sale_SK INT(11)
salesQuarter INT(11)
salesYear INT(11)
Indexes
orderdate_dim
Order_Date_SK INT(11)
OrderDate_NK DATE
OrderYear INT(4)
OrderQuarter INT(11)
OrderMonth INT(11)
OrderWeek INT(11)
OrderDay INT(11)
OrderFiscalYear INT(11)
OrderFiscalQuarter INT(11)
OrderFiscalMonth INT(11)
OrderFiscalWeek INT(11)
DayOfWeek INT(11)
Indexes
payment_order_shipping_junk_dim
Payment_Order_Shipping_Junk_SK INT(11)
ShippingMethod VARCHAR(45)
PaymentMethod VARCHAR(45)
OrderMethod VARCHAR(50)
Version INT
EffectiveDate DATE
ExpiryDate DATE
Indexes
sales_fact
Order_Date_SK INT(11)
Sales_Date_SK INT(11)
Product_SK INT(11)
Payment_Order_Shipping_JunkSK INT(11)
Customer_SK INT(11)
Amount DECIMAL(10,2)
Quantity INT(11)
ShipCost DECIMAL(10,2)
Discounted INT(11)
InvoiceID_DD INT(11)
Profit DECIMAL(10,2)
Number_of_days_to_ship INT(11)
Indexes
saledate_dim
SalesDate_SK INT(11)
SalesDate_NK DATE
SalesYear INT(11)
SalesQuarter INT(11)
SalesMonth INT(11)
SalesWeek INT(11)
SalesDay INT(11)
DayOfWeek INT(11)
SalesFiscalYear INT(11)
SalesFiscalQuarter INT(11)
SalesFiscalMonth INT(11)
SalesFiscalWeek INT(11)
Indexes
customer_dim
Customer_SK INT(11)
CustID_NK INT(11)
CustomerName VARCHAR(100)
Addr1 VARCHAR(50)
Addr2 VARCHAR(50)
City VARCHAR(50)
State VARCHAR(50)
Zip CHAR(5)
CustTypeID VARCHAR(20)
TypeName VARCHAR(50)
DivisionID INT(11)
CustomerDateFrom DATE
CustomerDateThrough DATE
CustomerVersionNumber INT
Indexes
collapsed_fact
Customer_SK INT(11)
Product_SK INT(11)
SalesDate_SK INT(11)
custID INT(11)
customerName VARCHAR(100)
SalesDate DATE
salesYear INT(11)
totalAmount DECIMAL(32,2)
totalQuantity DECIMAL(32,0)
ShipCOst DECIMAL(10,2)
DivisionID INT(11)
ProductID INT(11)
ProductName VARCHAR(50)
Indexes
sales_by_date_fact
SalesDate_SK INT(11)
SalesDate DATE
TotalAmount DECIMAL(32,2)
TotalQuantity DECIMAL(32,0)
Indexes

VIII. Handling Slowly Changing Dimensions (SCD)

We have performed the Slowly Changing Dimensions taking samples from two Dimension tables: Product and Customer.

On Product Dimension we performed SCD1,SCD2 and SCD6.

On Customer Dimension we performed SCD1 and SCD2.

The Sample size for the tables is as follows:

Product_Dimension = 114 rows

Transactional_Product = 34 rows

Customer_Dimension= 96 rows

Transactional_Customer = 25 rows

Product:

SCD Type 1:

This type of slowly changing dimension is used when there is an error in the data entry and we need to correct it by replacing the error with the right value/text.

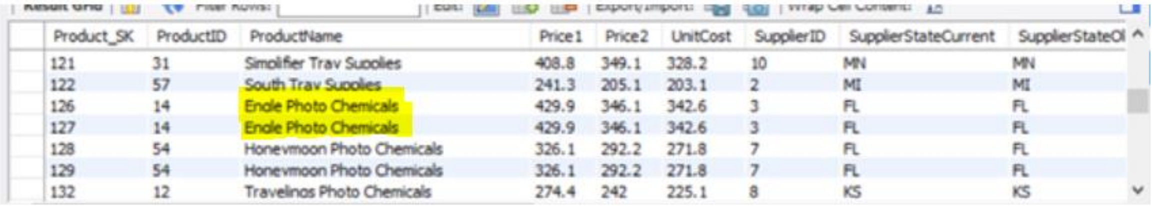
The erroneous record is updated by overwriting the incorrect record. Since the record is an error we do not keep an account of the old record and it is lost when the data is updated.

Example1:

In this example, we have changed the **ProductName** ‘Engle Photo Chemicals’ at ProductID :14 to ProductName ‘Eagle Photo Chemicals’ Considering assumption that there would have been a spelling mistake in this ProductName in a real scenario.

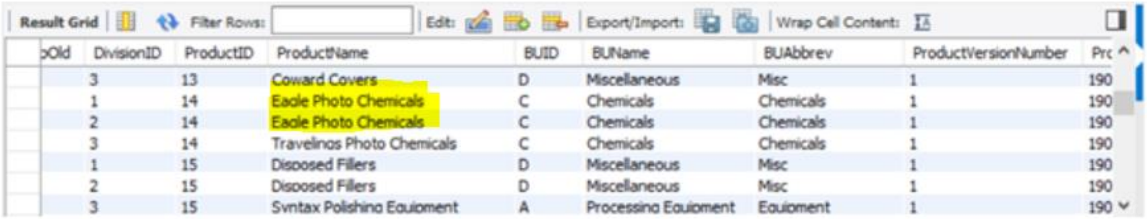
We performed the following in Pentaho using Dimension lookup/Update and selecting Punch Through for the attribute ‘**ProductName**’

Input



Product_SK	ProductID	ProductName	Price1	Price2	UnitCost	SupplierID	SupplierStateCurrent	SupplierStateOl
121	31	Simplifier Trav Supplies	408.8	349.1	328.2	10	MN	MN
122	57	South Trav Supplies	241.3	205.1	203.1	2	MI	MI
126	14	Engle Photo Chemicals	429.9	346.1	342.6	3	FL	FL
127	14	Engle Photo Chemicals	429.9	346.1	342.6	3	FL	FL
128	54	Honeymoon Photo Chemicals	326.1	292.2	271.8	7	FL	FL
129	54	Honeymoon Photo Chemicals	326.1	292.2	271.8	7	FL	FL
132	12	Travelings Photo Chemicals	274.4	242	225.1	8	KS	KS

Output

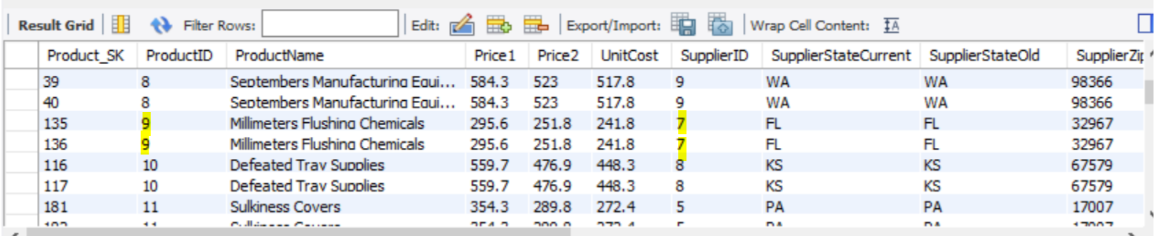


oOld	DivisionID	ProductID	ProductName	BUID	BUName	BUAbbrev	ProductVersionNumber	Prc
	3	13	Coward Covers	D	Miscellaneous	Misc	1	190
	1	14	Eagle Photo Chemicals	C	Chemicals	Chemicals	1	190
	2	14	Eagle Photo Chemicals	C	Chemicals	Chemicals	1	190
	3	14	Travelings Photo Chemicals	C	Chemicals	Chemicals	1	190
	1	15	Disposed Fillers	D	Miscellaneous	Misc	1	190
	2	15	Disposed Fillers	D	Miscellaneous	Misc	1	190
	3	15	Syntax Polishing Equipment	A	Processing Equipment	Equipment	1	190

Example 2:

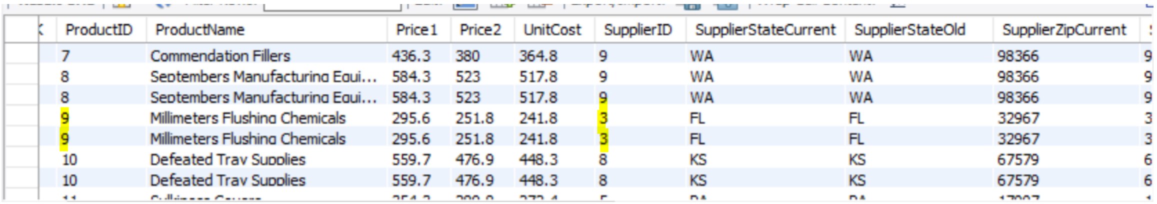
In the example below, we overwrite the **SupplierID** for ProductID 9 from 7 to 3.

Input:



	Product_SK	ProductID	ProductName	Price1	Price2	UnitCost	SupplierID	SupplierStateCurrent	SupplierStateOld	SupplierZi
	39	8	Septembers Manufacturing Equi...	584.3	523	517.8	9	WA	WA	98366
	40	8	Septembers Manufacturing Equi...	584.3	523	517.8	9	WA	WA	98366
	135	9	Millimeters Flushing Chemicals	295.6	251.8	241.8	7	FL	FL	32967
	136	9	Millimeters Flushing Chemicals	295.6	251.8	241.8	7	FL	FL	32967
	116	10	Defeated Tray Supplies	559.7	476.9	448.3	8	KS	KS	67579
	117	10	Defeated Tray Supplies	559.7	476.9	448.3	8	KS	KS	67579
	181	11	Sulkiness Covers	354.3	289.8	272.4	5	PA	PA	17007

Output:



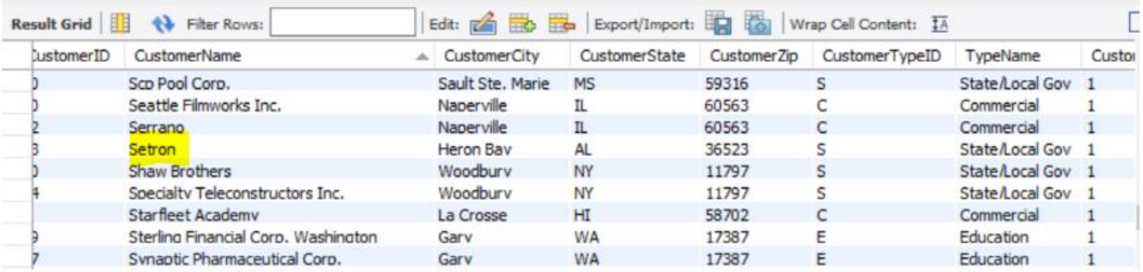
	ProductID	ProductName	Price1	Price2	UnitCost	SupplierID	SupplierStateCurrent	SupplierStateOld	SupplierZipCurrent	
	7	Commendation Fillers	436.3	380	364.8	9	WA	WA	98366	9
	8	Septembers Manufacturing Equi...	584.3	523	517.8	9	WA	WA	98366	9
	8	Septembers Manufacturing Equi...	584.3	523	517.8	9	WA	WA	98366	9
	9	Millimeters Flushing Chemicals	295.6	251.8	241.8	3	FL	FL	32967	3
	9	Millimeters Flushing Chemicals	295.6	251.8	241.8	3	FL	FL	32967	3
	10	Defeated Tray Supplies	559.7	476.9	448.3	8	KS	KS	67579	6
	10	Defeated Tray Supplies	559.7	476.9	448.3	8	KS	KS	67579	6

Customer:

Example3:

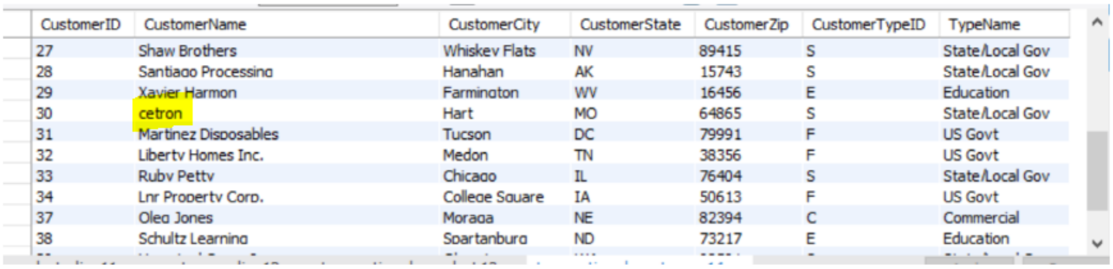
In this example We have the Overwritten the **CustomerName** ‘Setron’ to ‘Cetron’.

Input:



ustomerID	CustomerName	CustomerCity	CustomerState	CustomerZip	CustomerTypeID	TypeName	Custo
0	Scp Pool Corp.	Sault Ste. Marie	MS	59316	S	State/Local Gov	1
0	Seattle Filmworks Inc.	Naperville	IL	60563	C	Commercial	1
2	Serrano	Naperville	IL	60563	C	Commercial	1
3	Setron	Heron Bay	AL	36523	S	State/Local Gov	1
0	Shaw Brothers	Woodbury	NY	11797	S	State/Local Gov	1
4	Specialty Teleconstructors Inc.	Woodbury	NY	11797	S	State/Local Gov	1
	Starfleet Academy	La Crosse	HI	58702	C	Commercial	1
9	Sterling Financial Corp. Washington	Gary	WA	17387	E	Education	1
7	Synaptic Pharmaceutical Corp.	Gary	WA	17387	E	Education	1

Output:



	CustomerID	CustomerName	CustomerCity	CustomerState	CustomerZip	CustomerTypeID	TypeName
	27	Shaw Brothers	Whiskey Flats	NV	89415	S	State/Local Gov
	28	Santiago Processing	Hanahan	AK	15743	S	State/Local Gov
	29	Xavier Harmon	Farmington	WV	16456	E	Education
	30	cetron	Hart	MO	64865	S	State/Local Gov
	31	Martinez Disposables	Tucson	DC	79991	F	US Govt
	32	Liberty Homes Inc.	Medon	TN	38356	F	US Govt
	33	Ruby Petty	Chicago	IL	76404	S	State/Local Gov
	34	Lnr Property Corp.	College Square	IA	50613	F	US Govt
	37	Olga Jones	Moraga	NE	82394	C	Commercial
	38	Schultz Learning	Spartanburg	ND	73217	E	Education

SCD Type 2:

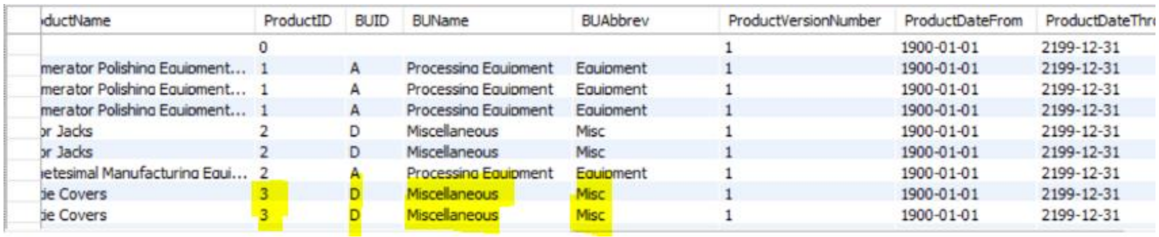
Product:

This type of SCD is used when there is a need to update the value of an attribute but at the same time keep a record of the old value for a non-erroneous change that has occured relevant to the business rules. In this case to show the current and old value we add a new row where the current value of the attribute is flagged by the column version number in addition to the effective and expiration dates (as Product_date_From and Product_date_through).

Example1:

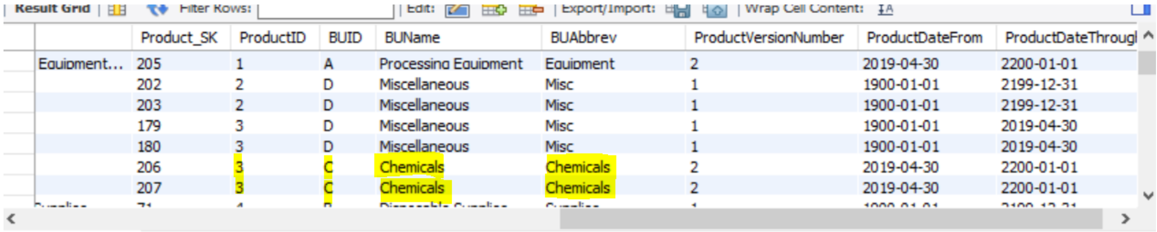
In this example we have updated **BUID**-D to C , **BUNAME** Miscellaneous to Chemicals and **BUAbbrev** Misc to Chemicals for the ProductID 3. We have also added the three-necessary column ProductVersionNumber, ProductDateFrom and ProductDateThrough.

Input:



	ıductName	ProductID	BUID	BUName	BUAbbrev	ProductVersionNumber	ProductDateFrom	ProductDateThr
		0				1	1900-01-01	2199-12-31
	merator Polishing Equipment...	1	A	Processing Equipment	Equipment	1	1900-01-01	2199-12-31
	merator Polishing Equipment...	1	A	Processing Equipment	Equipment	1	1900-01-01	2199-12-31
	merator Polishing Equipment...	1	A	Processing Equipment	Equipment	1	1900-01-01	2199-12-31
	or Jacks	2	D	Miscellaneous	Misc	1	1900-01-01	2199-12-31
	or Jacks	2	D	Miscellaneous	Misc	1	1900-01-01	2199-12-31
	ıetesimal Manufacturing Equi...	2	A	Processing Equipment	Equipment	1	1900-01-01	2199-12-31
	tie Covers	3	D	Miscellaneous	Misc	1	1900-01-01	2199-12-31
	tie Covers	3	D	Miscellaneous	Misc	1	1900-01-01	2199-12-31

Output:



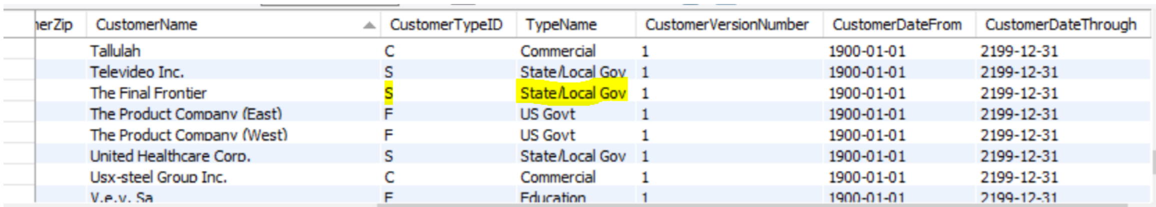
	Product_SK	ProductID	BUID	BUName	BUAbbrev	ProductVersionNumber	ProductDateFrom	ProductDateThrougl
Equipment...	205	1	A	Processing Equipment	Equipment	2	2019-04-30	2200-01-01
	202	2	D	Miscellaneous	Misc	1	1900-01-01	2199-12-31
	203	2	D	Miscellaneous	Misc	1	1900-01-01	2199-12-31
	179	3	D	Miscellaneous	Misc	1	1900-01-01	2019-04-30
	180	3	D	Miscellaneous	Misc	1	1900-01-01	2019-04-30
	206	3	C	Chemicals	Chemicals	2	2019-04-30	2200-01-01
	207	3	C	Chemicals	Chemicals	2	2019-04-30	2200-01-01

Customer:

Example2:

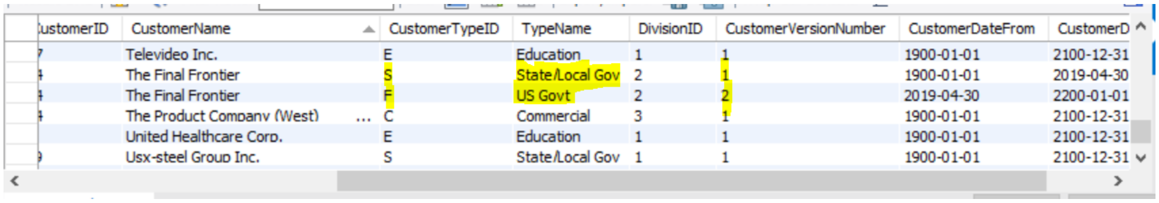
In this example, we have changed **CustTypeID** ‘S’ to ‘F’ and **TypeName** ‘State/Local Gov’ to ‘US Gov’ for CustomerName The Final Frontier by adding new row.

Input:



ıerZip	CustomerName	CustomerTypeID	TypeName	CustomerVersionNumber	CustomerDateFrom	CustomerDateThrough
	Tallulah	C	Commercial	1	1900-01-01	2199-12-31
	Televideo Inc.	S	State/Local Gov	1	1900-01-01	2199-12-31
	The Final Frontier	S	State/Local Gov	1	1900-01-01	2199-12-31
	The Product Company (East)	F	US Govt	1	1900-01-01	2199-12-31
	The Product Company (West)	F	US Govt	1	1900-01-01	2199-12-31
	United Healthcare Corp.	S	State/Local Gov	1	1900-01-01	2199-12-31
	Usx-steel Group Inc.	C	Commercial	1	1900-01-01	2199-12-31
	V.e.v. Sa	E	Education	1	1900-01-01	2199-12-31

Output:



ustomerID	CustomerName	CustomerTypeID	TypeName	DivisionID	CustomerVersionNumber	CustomerDateFrom	CustomerD
7	Televideo Inc.	E	Education	1	1	1900-01-01	2100-12-31
4	The Final Frontier	S	State/Local Gov	2	1	1900-01-01	2019-04-30
4	The Final Frontier	F	US Govt	2	2	2019-04-30	2200-01-01
4	The Product Company (West) ...	C	Commercial	3	1	1900-01-01	2100-12-31
	United Healthcare Corp.	E	Education	1	1	1900-01-01	2100-12-31
9	Usx-steel Group Inc.	S	State/Local Gov	1	1	1900-01-01	2100-12-31

SCD Type 6:

Product:

SCD 6 is used when we want to update a value of an attribute and also keep a track of the previous value in the same record. The updated value is incorporated by adding new row to the table using SCD2 .The Previous value of the attribute is accounted for by adding a new column using SCD3 which is overwritten using SCD1.

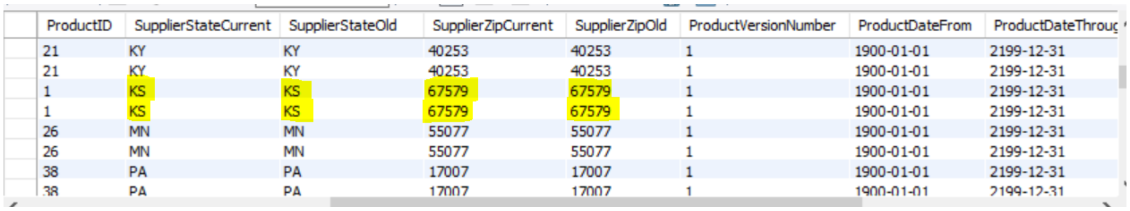
Thus SCD6 =SCD1+SCD2+SCD3.

Example1:

In this example we have changed **SupplierState** KS to WA which would also naturally lead to a change in the **SupplierZip** for the ProductID : 1

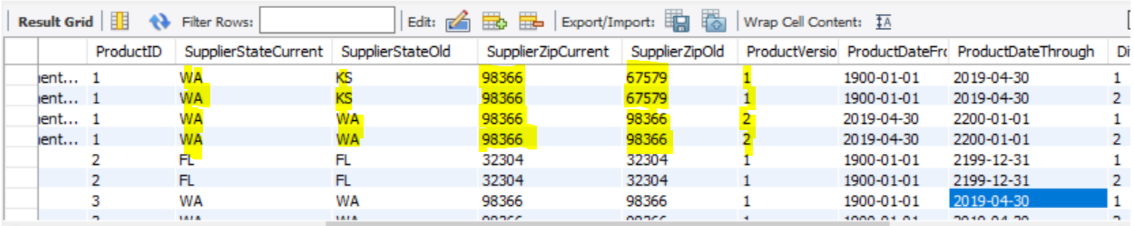
We can see below in the screenshots, SupplierStateCurrent, SupplierStateOld, SupplierZipCurrent, SupplierZipOld before and after applying the SCD's.

Input



ProductID	SupplierStateCurrent	SupplierStateOld	SupplierZipCurrent	SupplierZipOld	ProductVersionNumber	ProductDateFrom	ProductDateThroug
21	KY	KY	40253	40253	1	1900-01-01	2199-12-31
21	KY	KY	40253	40253	1	1900-01-01	2199-12-31
1	KS	KS	67579	67579	1	1900-01-01	2199-12-31
1	KS	KS	67579	67579	1	1900-01-01	2199-12-31
26	MN	MN	55077	55077	1	1900-01-01	2199-12-31
26	MN	MN	55077	55077	1	1900-01-01	2199-12-31
38	PA	PA	17007	17007	1	1900-01-01	2199-12-31
38	PA	PA	17007	17007	1	1900-01-01	2199-12-31

Output:



	ProductID	SupplierStateCurrent	SupplierStateOld	SupplierZipCurrent	SupplierZipOld	ProductVersio	ProductDateFr	ProductDateThrough	Di
ient...	1	WA	KS	98366	67579	1	1900-01-01	2019-04-30	1
ient...	1	WA	KS	98366	67579	1	1900-01-01	2019-04-30	2
ient...	1	WA	WA	98366	98366	2	2019-04-30	2200-01-01	1
ient...	1	WA	WA	98366	98366	2	2019-04-30	2200-01-01	2
	2	FL	FL	32304	32304	1	1900-01-01	2199-12-31	1
	2	FL	FL	32304	32304	1	1900-01-01	2199-12-31	2
	3	WA	WA	98366	98366	1	1900-01-01	2019-04-30	1

IX. Many-to-Many (N-M) Relationship Implementation Option

Based on the data we are under the assumption that fact table contains supplier's information which has one shipping company related to it. But in reality, one supplier can have multiple shipping companies related to it. In the previous Data mart, we can calculate the total ship cost by joining fact and supplier, but that model does not allow to analyze sales by single supplier's shipping company.

There are various ways to solve this:

1) **Bridge table Method:**

   The bridge table is an intersection between suppliers and shipping companies. This approach is similar to solving many to many entities in database with the only difference that this table has a weighted factor associated to it. The weighting factor denotes the weight or percentage that identifies the contribution of a shipping company in delivering an order for a supplier. This is important because two shipping companies can be responsible for completing one order. The weighted factor which totals to 1 per one order shipment is distributed reasonably among the participating shipping companies. This method also uses a group key to illustrate all the possible one to one shipping and supplier combinations

2) **Boolean Column Method**:

   The Boolean method is creating a column for each possible value of shipping company in the supplier table.

3) **Multiple Column Method :**

The multiple column method consists of having columns for the number of choices between shippers and suppliers. This has limitations because it is tightly coupled to the application; but is easily transformed.

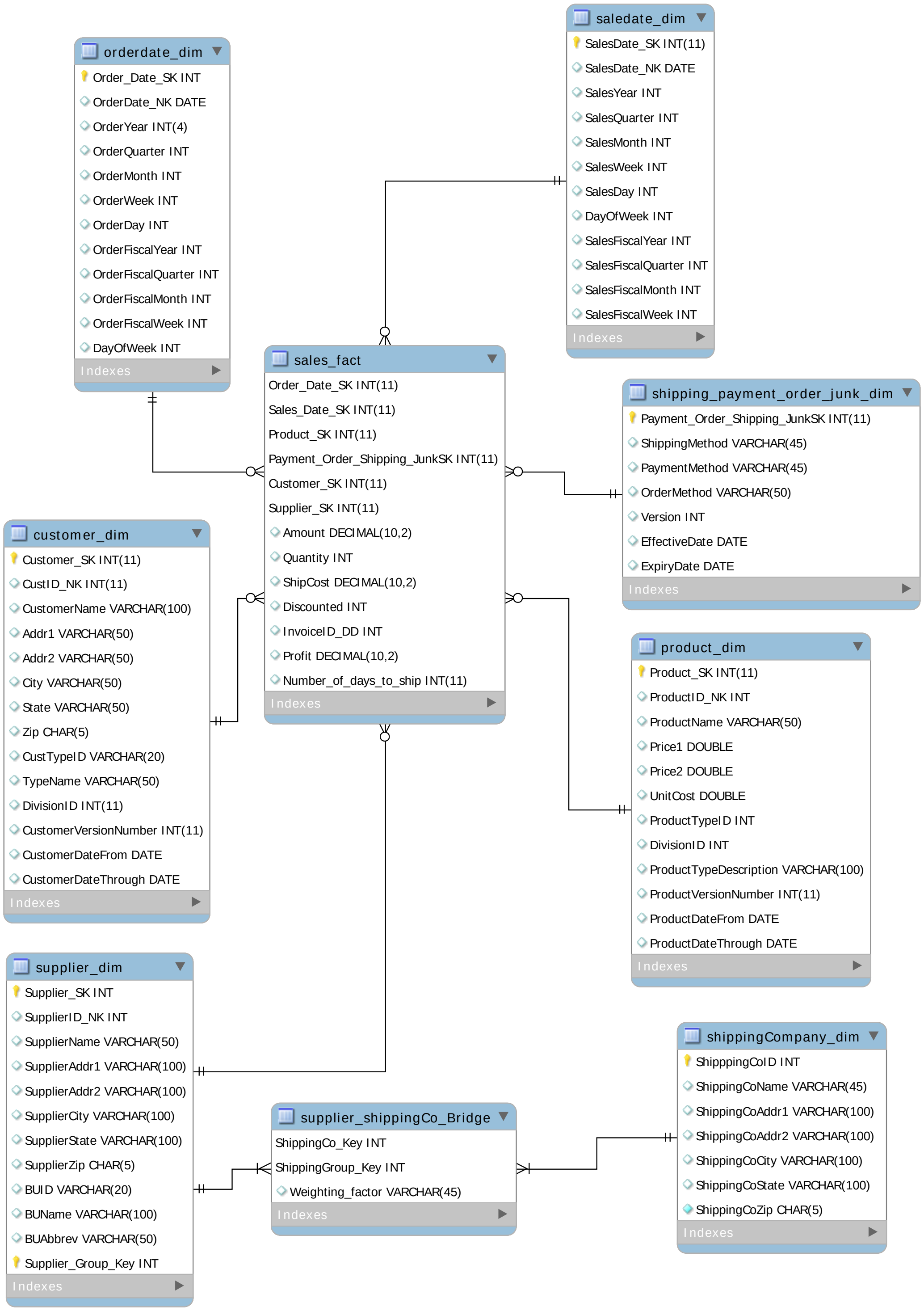
Out of these methods, **Boolean and Bridge** methods are superior. However, when we have more than 100 values in dimensions(Shipping and Supplier), creating Booleans will take a lot of time. So, keeping this in mind, we propose **Bridge method as the best approach to solve N-M problem**.

IMPLEMENTATION:

1) Earlier Data mart had supplier included in product dimension. So, we create a new supplier Dimension, assign a Surrogate Key to each record and attach it to fact table with a Supplier_SK Foreign key.
2) We create a shipping dimension as well to list all the shipping companies that can deliver products for each supplier.
3) We need a group key as well; Group key represents a Supplier and shipping group. For example, Group 1 = Supplier 1, Shipping Company 1 And Group 2 can be = Supplier 2, Shipping Company 1 and Shipping Company 2.
4) This group key is included in the supplier dimension.
5) A bridge table is created, which has the Group key, Shipping SK and Weighted factor.
6) Weighted factor can be assigned by discussing with domain experts. As multiple shipping company can fulfill parts of an order, they can be given weights for the part of the order they are fulfilling. For example - shipment fulfilled by Shipping Company 1 from

Company warehouse to location A can be weighted as 0.2 and shipment fulfilled by Shipping Company 2 from Company warehouse to location A can be weighted as 0.8. This can depend on shipping methods as well i.e. if the order is shipped by air , truck or train.

7) With the help of this weighted factor, we can analyze sales fulfilled by different shipping companies for a particular supplier.


orderdate_dim
Order_Date_SK INT
OrderDate_NK DATE
OrderYear INT(4)
OrderQuarter INT
OrderMonth INT
OrderWeek INT
OrderDay INT
OrderFiscalYear INT
OrderFiscalQuarter INT
OrderFiscalMonth INT
OrderFiscalWeek INT
DayOfWeek INT
Indexes
saledate_dim
SalesDate_SK INT(11)
SalesDate_NK DATE
SalesYear INT
SalesQuarter INT
SalesMonth INT
SalesWeek INT
SalesDay INT
DayOfWeek INT
SalesFiscalYear INT
SalesFiscalQuarter INT
SalesFiscalMonth INT
SalesFiscalWeek INT
Indexes
sales_fact
Order_Date_SK INT(11)
Sales_Date_SK INT(11)
Product_SK INT(11)
Payment_Order_Shipping_JunkSK INT(11)
Customer_SK INT(11)
Supplier_SK INT(11)
Amount DECIMAL(10,2)
Quantity INT
ShipCost DECIMAL(10,2)
Discounted INT
InvoiceID_DD INT
Profit DECIMAL(10,2)
Number_of_days_to_ship INT(11)
Indexes
shipping_payment_order_junk_dim
Payment_Order_Shipping_JunkSK INT(11)
ShippingMethod VARCHAR(45)
PaymentMethod VARCHAR(45)
OrderMethod VARCHAR(50)
Version INT
EffectiveDate DATE
ExpiryDate DATE
Indexes
customer_dim
Customer_SK INT(11)
CustID_NK INT(11)
CustomerName VARCHAR(100)
Addr1 VARCHAR(50)
Addr2 VARCHAR(50)
City VARCHAR(50)
State VARCHAR(50)
Zip CHAR(5)
CustTypeID VARCHAR(20)
TypeName VARCHAR(50)
DivisionID INT(11)
CustomerVersionNumber INT(11)
CustomerDateFrom DATE
CustomerDateThrough DATE
Indexes
product_dim
Product_SK INT(11)
ProductID_NK INT
ProductName VARCHAR(50)
Price1 DOUBLE
Price2 DOUBLE
UnitCost DOUBLE
ProductTypeID INT
DivisionID INT
ProductTypeDescription VARCHAR(100)
ProductVersionNumber INT(11)
ProductDateFrom DATE
ProductDateThrough DATE
Indexes
supplier_dim
Supplier_SK INT
SupplierID_NK INT
SupplierName VARCHAR(50)
SupplierAddr1 VARCHAR(100)
SupplierAddr2 VARCHAR(100)
SupplierCity VARCHAR(100)
SupplierState VARCHAR(100)
SupplierZip CHAR(5)
BUID VARCHAR(20)
BUName VARCHAR(100)
BUAbbrev VARCHAR(50)
Supplier_Group_Key INT
Indexes
shippingCompany_dim
ShipppingCoID INT
ShippingCoName VARCHAR(45)
ShippingCoAddr1 VARCHAR(100)
ShippingCoAddr2 VARCHAR(100)
ShippingCoCity VARCHAR(100)
ShippingCoState VARCHAR(100)
ShippingCoZip CHAR(5)
Indexes
supplier_shippingCo_Bridge
ShippingCo_Key INT
ShippingGroup_Key INT
Weighting_factor VARCHAR(45)
Indexes

References:

[1]
II-Yeol Song,Edward Ewen,William rowen,carl Medsker (2001), “An analysis of many-to-many relationships between fact and dimension tables in dimensional modeling”,Proceedings of the International, Retrieved From :https://www.academia.edu/977976/An_analysis_of_many-to-many_relationships_between_fact_and_dimension_tables_in_dimensional_modeling

[2] Packt (2009,December 28), Solving Many-Many Relationships in Dimensional Modeling,
Retrived from: https://hub.packtpub.com/solving-many-many-relationship-dimensional-modeling

[3]
Nicome,(2017, September 13),Data Modeling Many-many Relationship [Web Blog Post],Retrived April 30, 2019 from
https://gerardnico.com/data/modeling/many-to-many#boolean_column_method

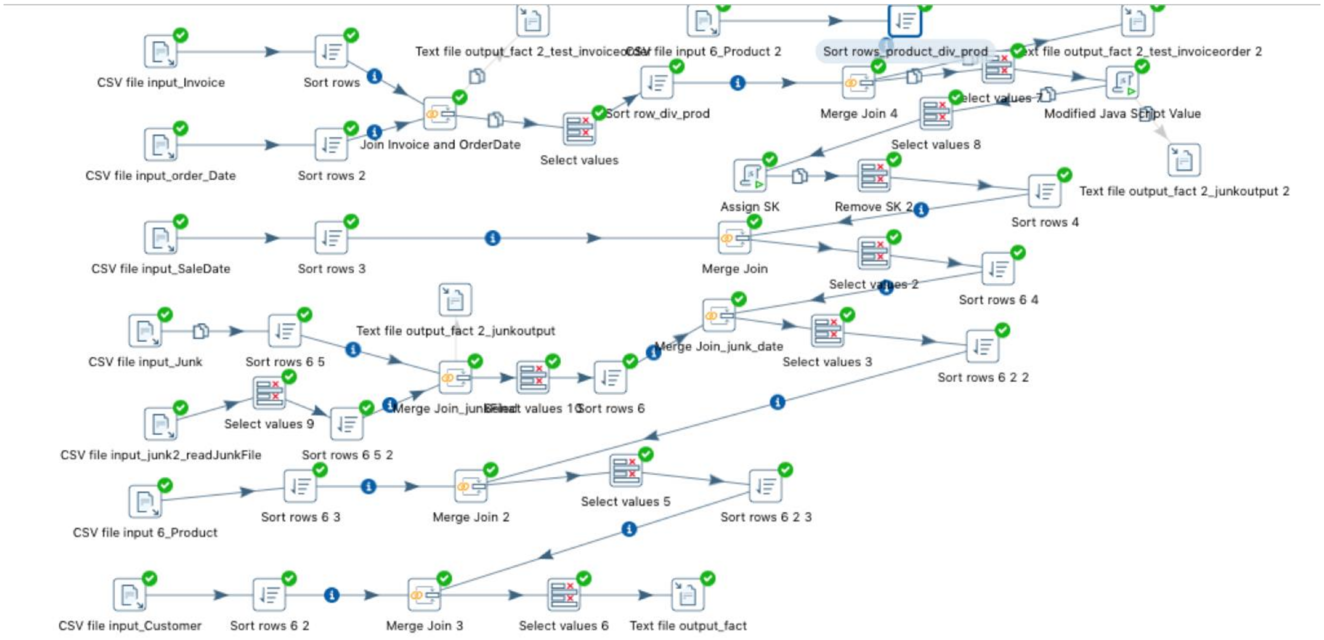
X. Appendix

1) Lab 3 Fixes:

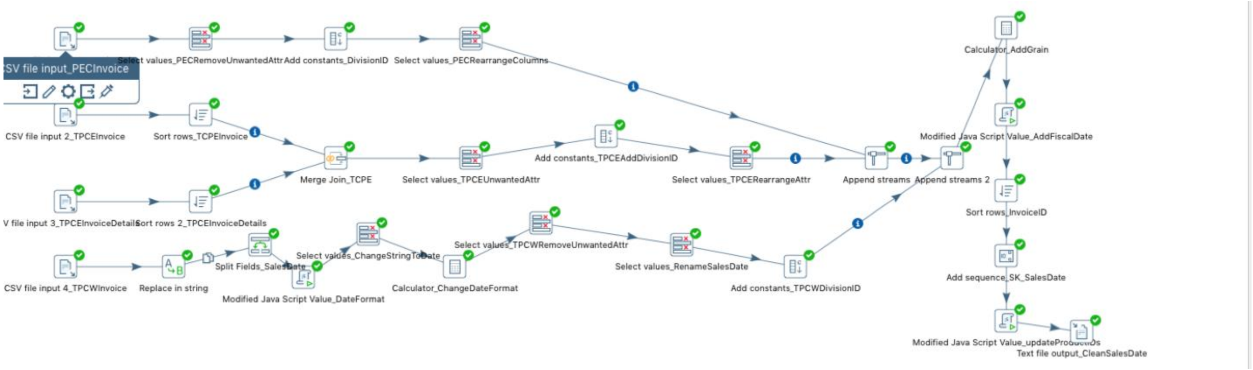
a) The Cleaned file for Product is updated. The product ID 33 is changes to 3.

b) Added version Number, Effective Date and Expiry Date for identified SCDs.

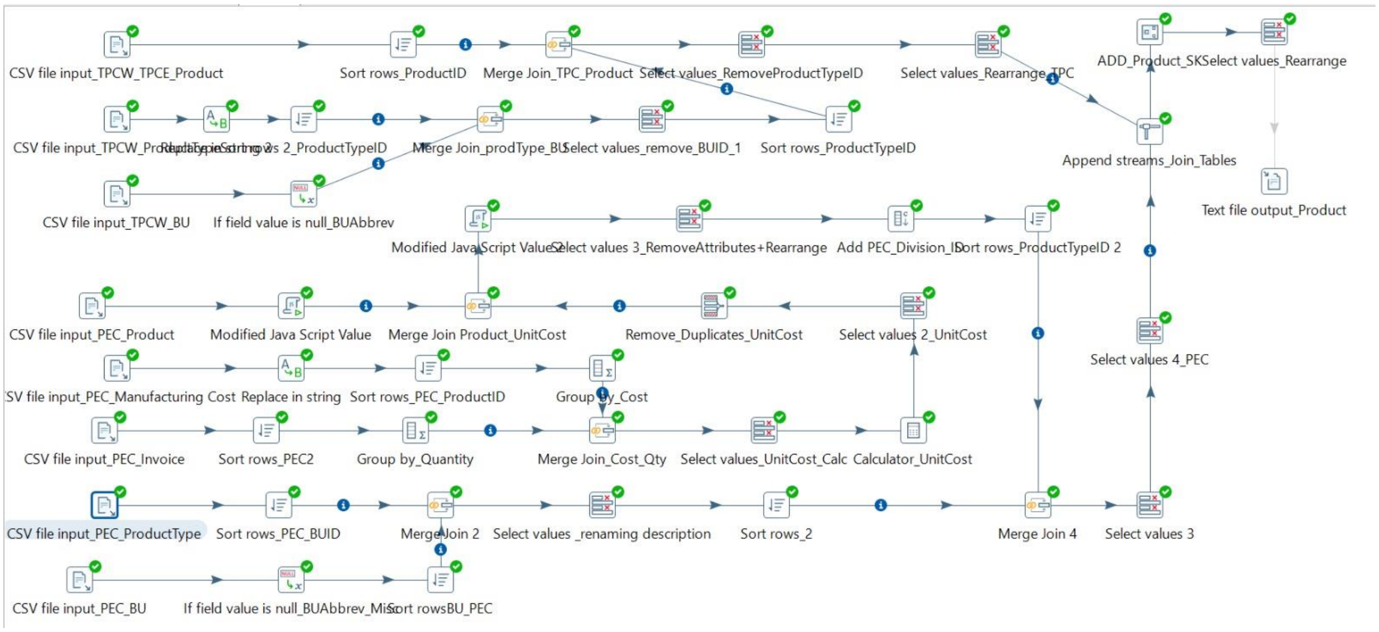
2) Customer Transformation



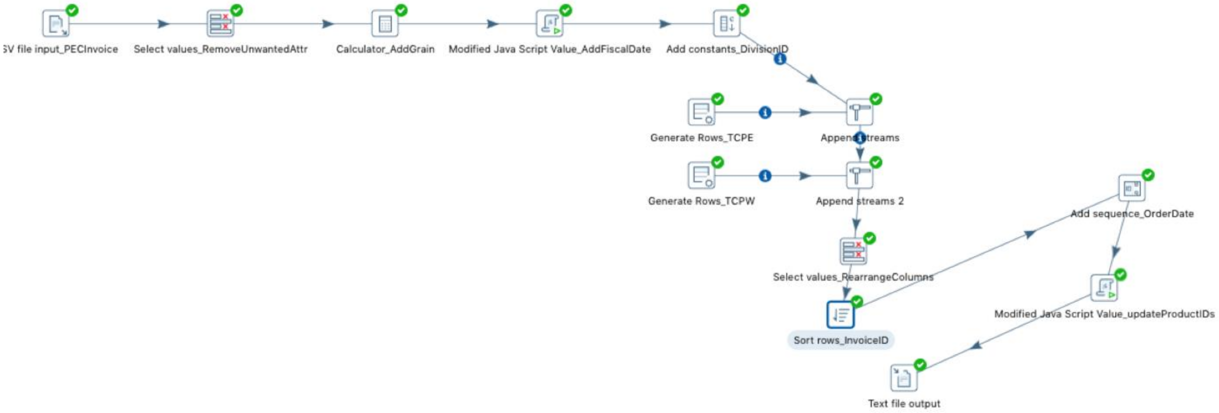
3) Sales Date Transformation


CSV file input_PECInvoice
Select values_PECRemoveUnwantedAttr
Add constants_DivisionID
Select values_PECRearrangeColumns
CSV file input 2_TPCEInvoice
Sort rows_TCPEInvoice
Merge Join_TCPE
Select values_TPCEUnwantedAttr
Add constants_TPCEAddDivisionID
Select values_TPCERearrangeAttr
Append streams
Append streams 2
Calculator_AddGrain
Modified Java Script Value_AddFiscalDate
Sort rows_InvoiceID
Add sequence_SK_SalesDate
Modified Java Script Value_updateProductIDs
Text file output_CleanSalesDate
V file input 3_TPCEInvoiceDetails
Sort rows 2_TPCEInvoiceDetails
CSV file input 4_TPCWInvoice
Replace in string
Split Fields_SalesDate
Modified Java Script Value_DateFormat
Select values_ChangeStringToDate
Calculator_ChangeDateFormat
Select values_TPCWRemoveUnwantedAttr
Select values_RenameSalesDate
Add constants_TPCWDivisionID

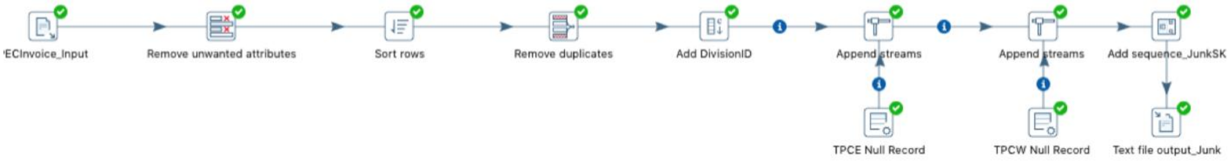
4) Product Transformation


CSV file input_TPCW_TPCE_Product
Sort rows_ProductID
Merge Join_TPC_Product
Select values_RemoveProductTypeID
Select values_Rearrange_TPC
ADD_Product_SK
Select values_Rearrange
CSV file input_TPCW_ProductType
Replace in string 2
Sort rows 2_ProductTypeID
Merge Join_prodType_BU
Select values_remove_BUID_1
Sort rows_ProductTypeID
Append streams_Join_Tables
Text file output_Product
CSV file input_TPCW_BU
If field value is null_BUAbbrev
Modified Java Script Value 2
Select values 3_RemoveAttributes+Rearrange
Add PEC_Division_ID
Sort rows_ProductTypeID 2
CSV file input_PEC_Product
Modified Java Script Value
Merge Join Product_UnitCost
Remove_Duplicates_UnitCost
Select values 2_UnitCost
Select values 4_PEC
CSV file input_PEC_Manufacturing Cost
Replace in string
Sort rows_PEC_ProductID
Group by_Cost
CSV file input_PEC_Invoice
Sort rows_PEC2
Group by_Quantity
Merge Join_Cost_Qty
Select values_UnitCost_Calc
Calculator_UnitCost
CSV file input_PEC_ProductType
Sort rows_PEC_BUID
Merge Join 2
Select values _renaming description
Sort rows_2
Merge Join 4
Select values 3
CSV file input_PEC_BU
If field value is null_BUAbbrev_Misc
Sort rowsBU_PEC

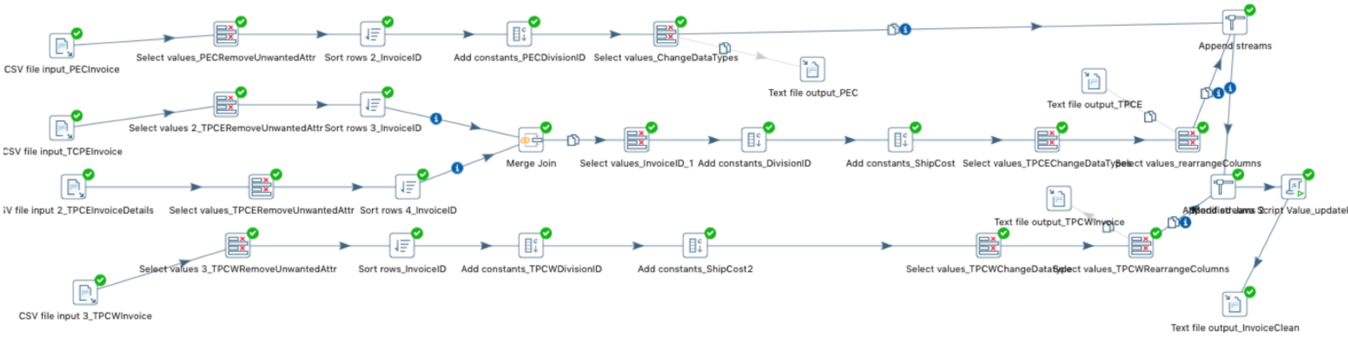
5) OrderDate Transformation



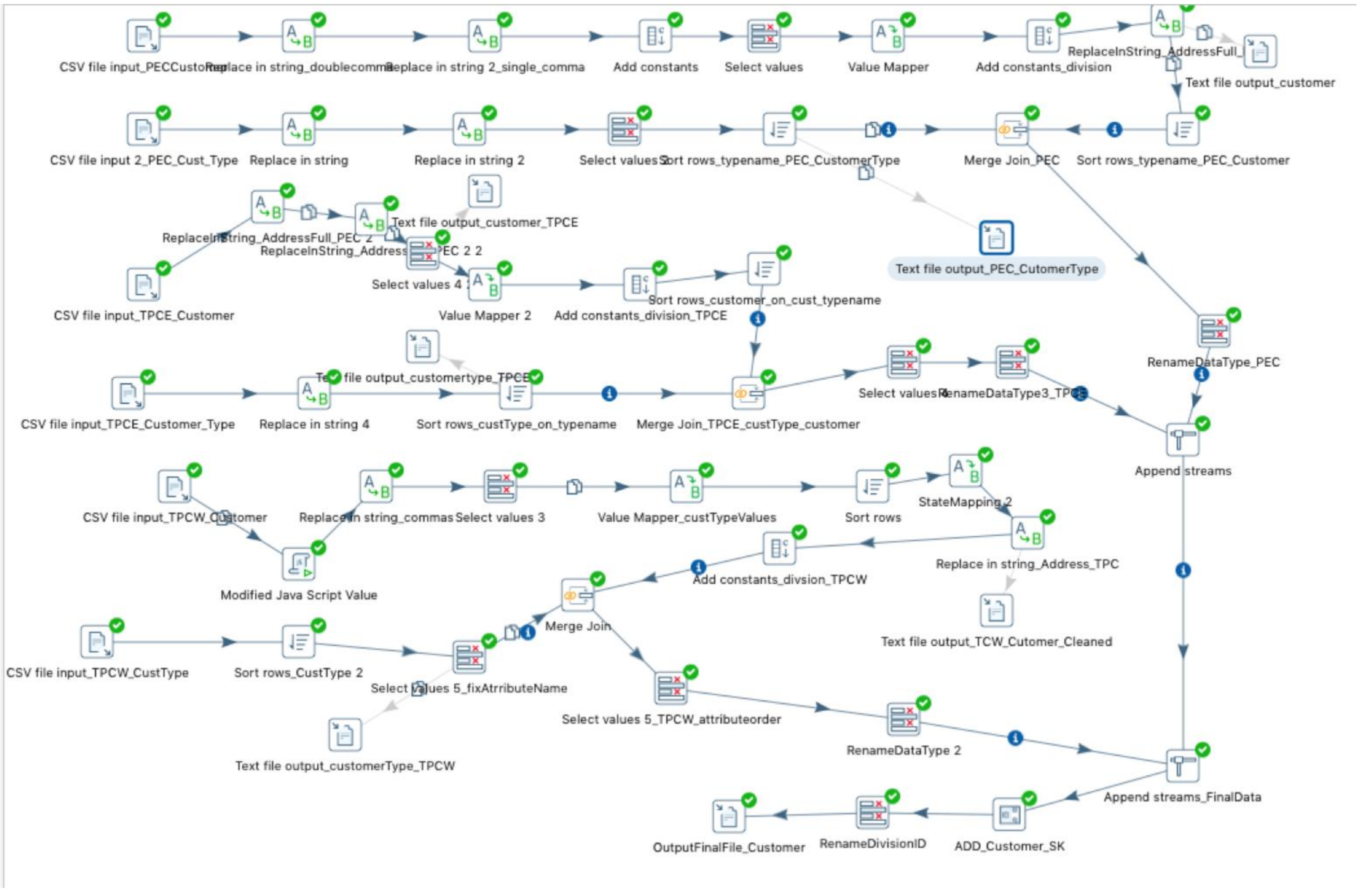
6) Junk Prep Transformation



7) Invoice Transformation


CSV file input_PECInvoice
Select values_PECRemoveUnwantedAttr
Sort rows 2_InvoiceID
Add constants_PECDivisionID
Select values_ChangeDataTypes
Text file output_PEC
Append streams
CSV file input_TCPEInvoice
Select values 2_TPCERemoveUnwantedAttr
Sort rows 3_InvoiceID
Merge Join
Select values_InvoiceID_1
Add constants_DivisionID
Add constants_ShipCost
Select values_TPCEChangeDataTypes
Select values_rearrangeColumns
Text file output_TPCE
;V file input 2_TPCEInvoiceDetails
Select values_TPCERemoveUnwantedAttr
Sort rows 4_InvoiceID
Append streams 2
Modified Java Script Value_updatel
Text file output_TPCWInvoice
Select values 3_TPCWRemoveUnwantedAttr
Sort rows_InvoiceID
Add constants_TPCWDivisionID
Add constants_ShipCost2
Select values_TPCWChangeDataTypes
Select values_TPCWRearrangeColumns
CSV file input 3_TPCWInvoice
Text file output_InvoiceClean

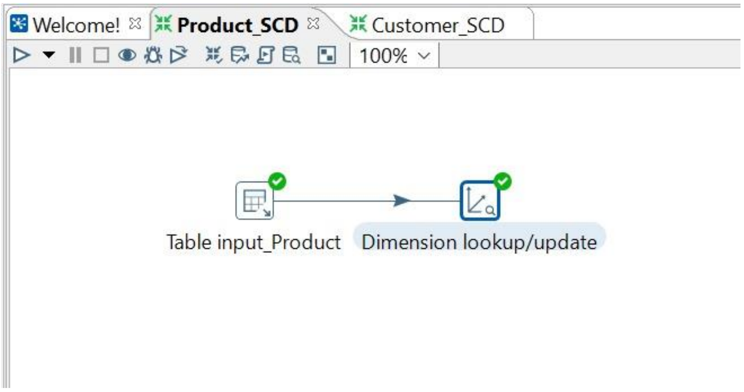
8) **Sales Fact Transformation**



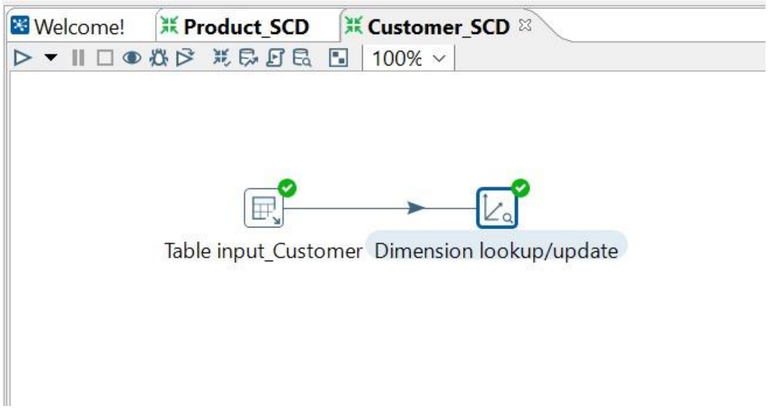
CSV file input_PECCustoRmeplace in string_doublecommaReplace in string 2_single_comma
Add constants
Select values
Value Mapper
Add constants_division
ReplaceInString_AddressFull_
Text file output_customer
CSV file input 2_PEC_Cust_Type
Replace in string
Replace in string 2
Select values Sort rows_typename_PEC_CustomerType
Merge Join_PEC
Sort rows_typename_PEC_Customer
ReplaceInString_AddressFull_PEC 2
ReplaceInString_Address_PEC 2 2
Text file output_customer_TPCE
Text file output_PEC_CutomerType
CSV file input_TPCE_Customer
Select values 4
Value Mapper 2
Add constants_division_TPCE
Sort rows_customer_on_cust_typename
RenameDataType_PEC
CSV file input_TPCE_Customer_Type
Replace in string 4
Text file output_customertype_TPCE
Sort rows_custType_on_typename
Merge Join_TPCE_custType_customer
Select values RenameDataType3_TPCE
Append streams
CSV file input_TPCW_Customer
Replace in string_commas
Select values 3
Value Mapper_custTypeValues
Sort rows
StateMapping 2
Modified Java Script Value
Add constants_divsion_TPCW
Replace in string_Address_TPC
Merge Join
Text file output_TCW_Cutomer_Cleaned
CSV file input_TPCW_CustType
Sort rows_CustType 2
Select values 5_fixAtrributeName
Select values 5_TPCW_attributeorder
RenameDataType 2
Text file output_customerType_TPCW
Append streams_FinalData
OutputFinalFile_Customer
RenameDivisionID
ADD_Customer_SK

9) **SCD**

On Product Dimension:



On Customer Dimension:


Welcome!
Product_SCD
Customer_SCD
100%
Table input_Customer
Dimension lookup/update