*The Product Company*

Data Mart Development Report

Contents

[I Data Mart Design Definition 1](#_Toc451192778)

[1 Universe of Discourse 1](#_Toc451192779)

[2 Information Package 1](#_Toc451192780)

[3 Entity Definitions 1](#_Toc451192781)

[II Dimensional Model 2](#_Toc451192782)

[III Data Staging: ETL – Data Extract File Definitions 3](#_Toc451192783)

[IV Data Staging: ETL – Source-to-Target Mappings 5](#_Toc451192784)

[1 Customer Dimension 5](#_Toc451192785)

[2 Junk Dimension 5](#_Toc451192786)

[3 Product Dimension 6](#_Toc451192787)

[4 Date Dimension 7](#_Toc451192788)

[5 Supplier Dimension 8](#_Toc451192789)

[6 Sales Fact 9](#_Toc451192790)

[V SQL Code – Tables & Constraints 10](#_Toc451192791)

[VI Data Staging Activities – ETL 14](#_Toc451192792)

[1 Data Cleansing 14](#_Toc451192793)

[2 Data Transformation 16](#_Toc451192794)

[3 Table Population 25](#_Toc451192795)

[VII End User Applications 28](#_Toc451192796)

[1 Queries 28](#_Toc451192797)

[2 Views & Summaries 29](#_Toc451192798)

[VIII Handling Slowly Changing Dimensions (SCD) 30](#_Toc451192799)

[IX Many-to-Many (N-M) Relationship Implementation Option 36](#_Toc451192800)

[X References 37](#_Toc451192801)

# Data Mart Design Definition

## Universe of Discourse

|  |
| --- |
| This data mart describes the sales of products between three divisions or others customers. |

## Information Package

Process Name : Product Sales

Grain : Each individual sales of product on a daily basis

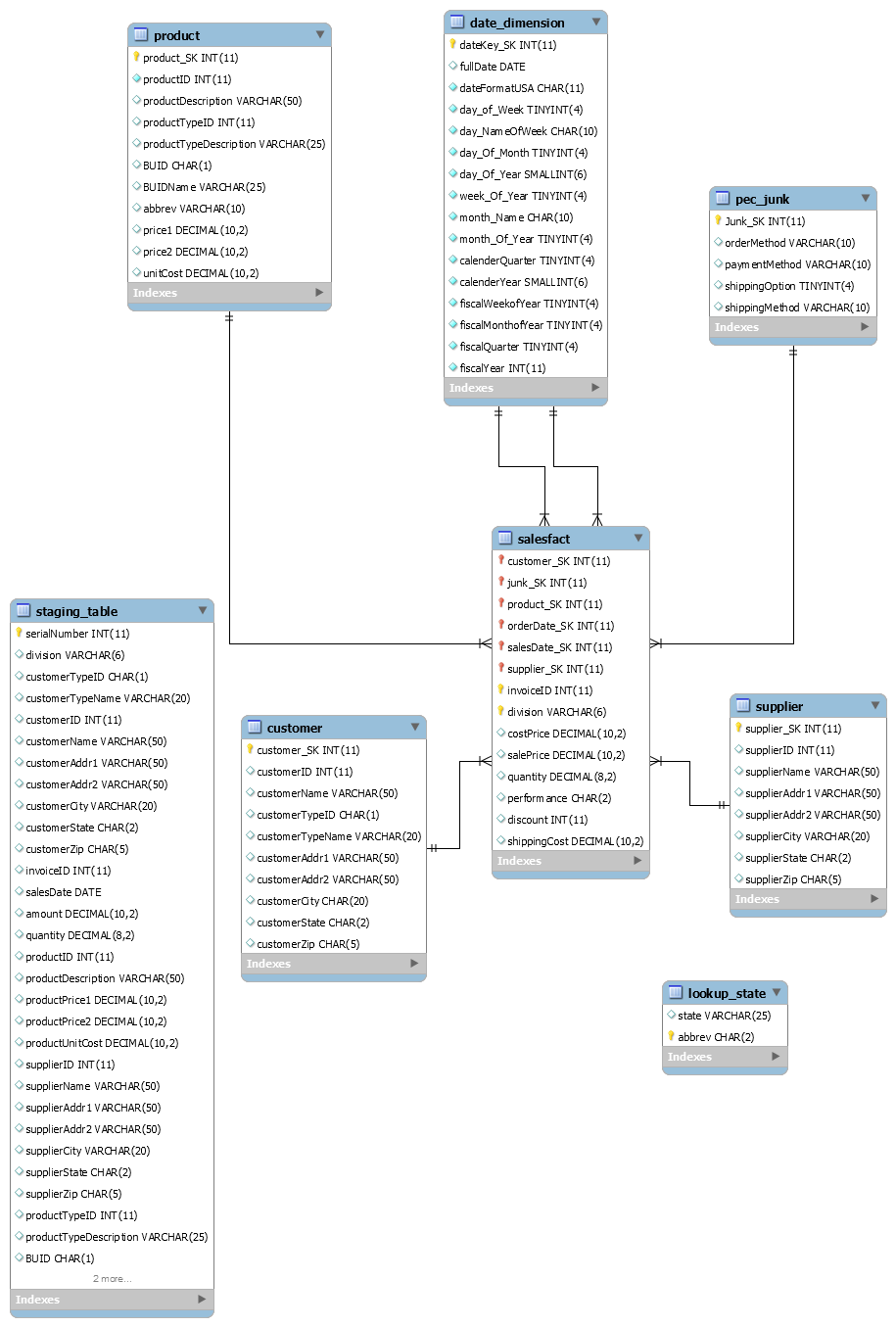
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Supplier | Product | customer | Date | PEC\_Junk |
| Name | ProductID | CustomerID | Calendar Year | ShippingMethod |
| SupplierID | Description | Name | Calendar Quarter | shippingOption |
| Address1 | price1 | custTypeID | Calendar Month | OrderMethod |
| Address2 | price2 | typeName | Calendar Week | PaymentMethod |
| City | unitCost | Address1 | Calendar Day |  |
| State | productTypeID | Address2 | Fiscal Year |  |
| Zipcode | typeDescription | City | Fiscal Quarter |  |
|  | BUID | State | Fiscal month |  |
|  | BUIDname | Zipcode | Fiscal week |  |
|  | Abbrev |  |  |  |

Facts: **Qty, salePrice, costPrice, Performance (Days between SaleDate and OrderDate), discount, shippingCost**

## Entity Definitions

|  |  |
| --- | --- |
| **Entity** | **Entity Definition** (*genus differentia*) |
| **Customer** | It provides details about the customer belonging to *The Product Company* |
| **Product** | It provides details about the product along with its cost sold by all the divisions (TPC-East, TPC-West and PEC) of *The Product Company.* |
| **Supplier** | It provides details about all the suppliers corresponding to *The Product Company.* |
| **Date** | It describes all the hierarchies of Date (calendar as well as fiscal) to help *The Product Company* to track their product saleDate and orderDate. |
| **PEC\_Junk** | It explains three methods to track the order which are shipment method, payment method and ordering method. |
| **Salesfact** | It consists of all the facts required to find the financial performance and reporting for *The Product Company.* |

# Dimensional Model

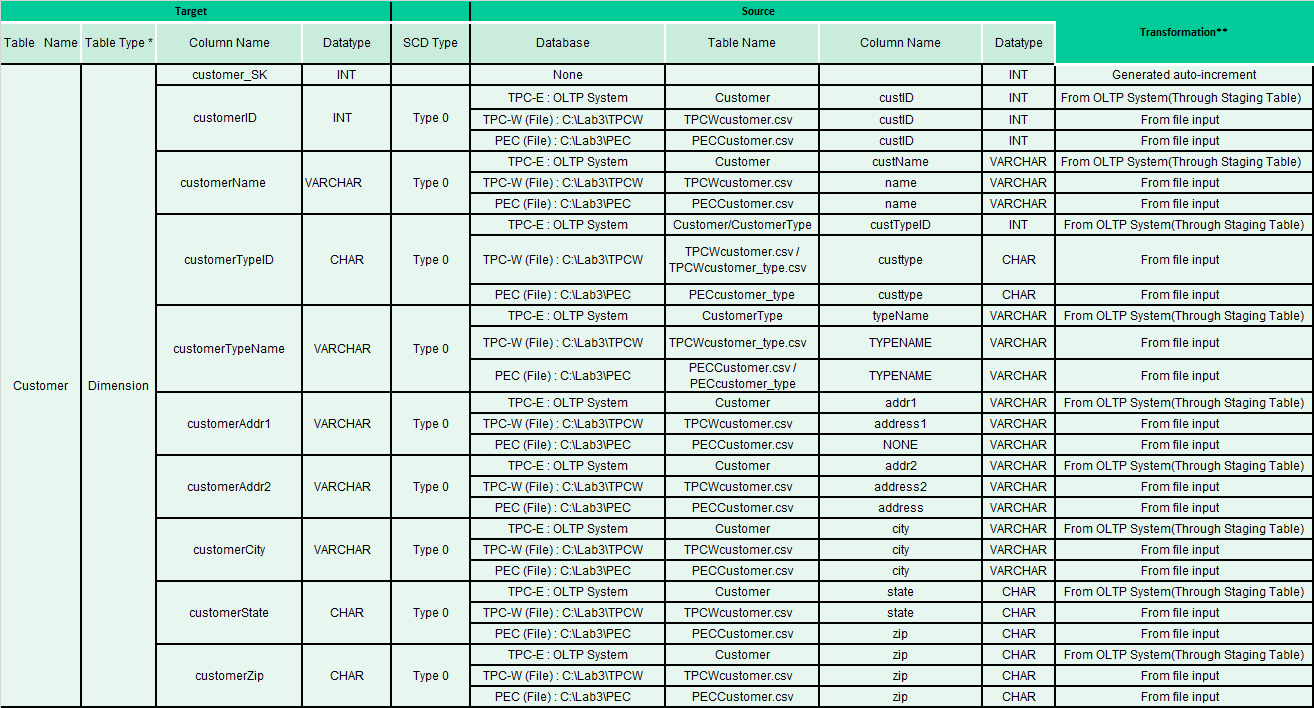


# Data Staging: ETL – Data Extract File Definitions

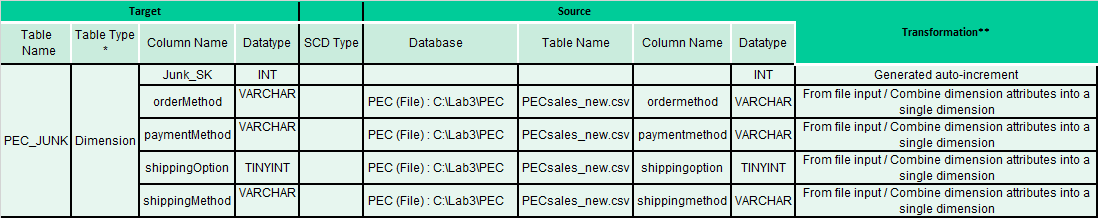
|  |  |  |  |
| --- | --- | --- | --- |
| **DM Table** | **Data Extract File(s)** | | **Format** |
| Customer | **TPC-E** | Customer | Table |
| CustomerType | Table |
| **TPC-W** | TPCWcustomer | File (.csv) |
| TPCWcustomer\_type | File (.csv) |
| **PEC** | PECcustomer | File (.csv) |
| PECcustomer\_type | File (.csv) |
| PEC\_Junk | **TPC-E** | None | None |
| **TPC-W** | None | None |
| **PEC** | PECsales\_new | File (.csv) |
| Product | **TPC-E** | Product | Table |
| ProdType | Table |
| BusinessUnit | Table |
| **TPC-W** | TPCWproduct | File (.csv) |
| TPCWproduct\_type | File (.csv) |
| TPCWbusiness\_unit | File (.csv) |
| **PEC** | PECproduct | File (.csv) |
| PECproduct\_type | File (.csv) |
| PECbusiness\_unit | File (.csv) |
| Supplier | **TPC-E** | Supplier | Table |
| **TPC-W** | Product | File (.csv) |
| **PEC** | PECproduct | File (.csv) |
| Sales Fact | **TPC-E** | InvoiceHdr | Table |
| InvoiceDtl | Table |
| **TPC-W** | TPCWsales\_new | File (.csv) |
| **PEC** | PECsales\_new | File (.csv) |

# Data Staging: ETL – Source-to-Target Mappings

## Customer Dimension



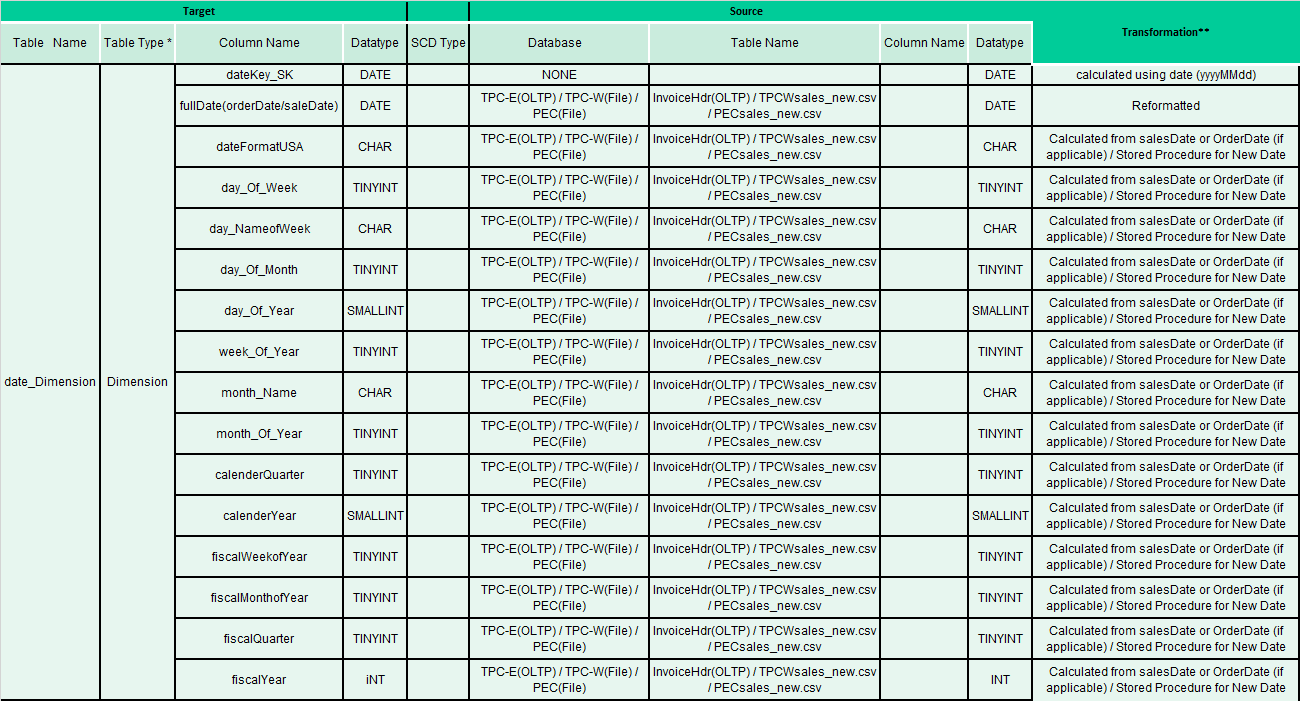
## Junk Dimension



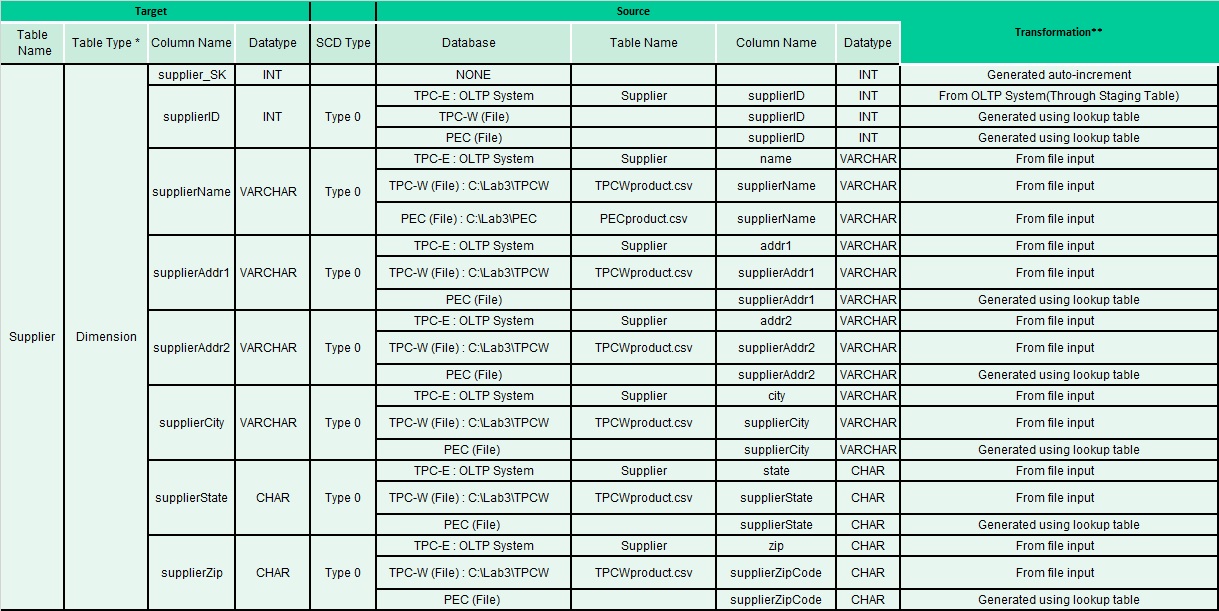
## Product Dimension



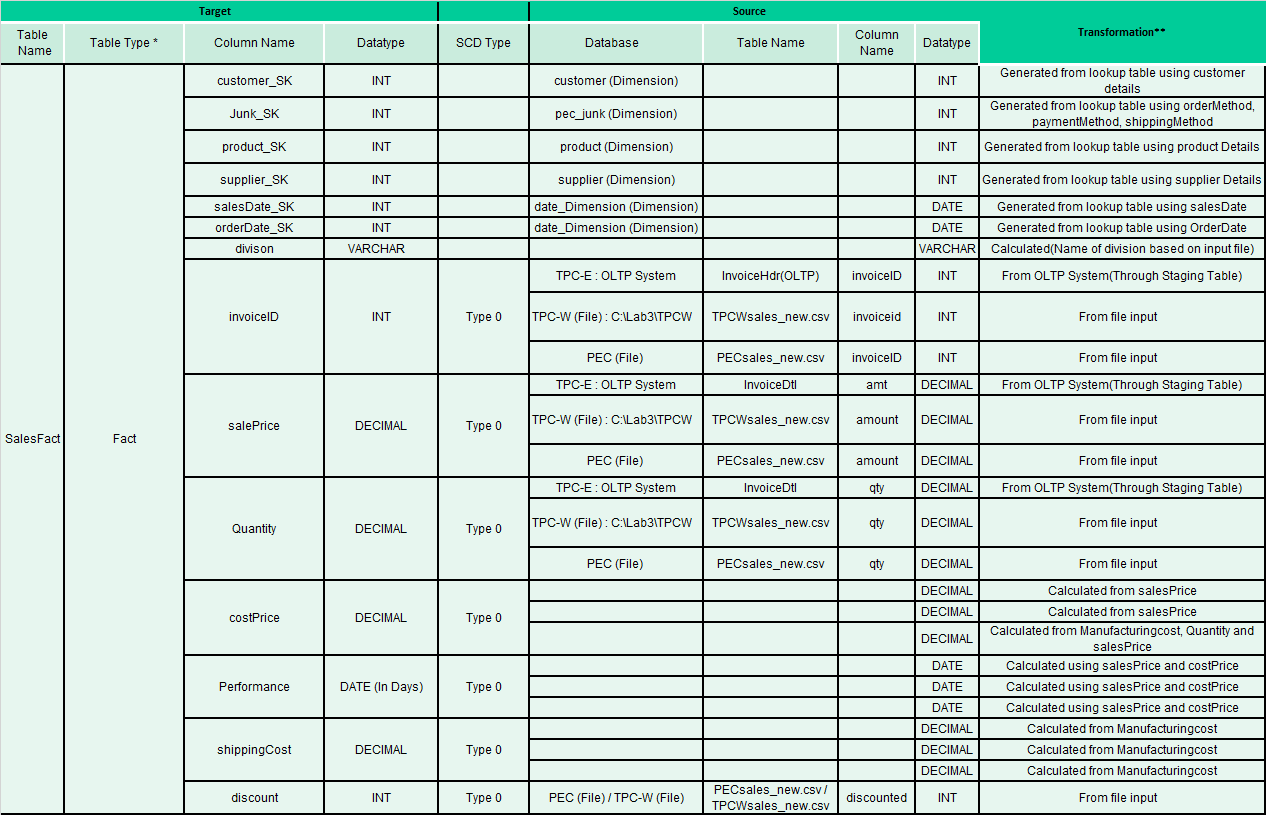
## Date Dimension



## Supplier Dimension



## Sales Fact



# SQL Code – Tables & Constraints

-- MySQL Workbench Forward Engineering

**SET** @OLD\_UNIQUE\_CHECKS**=**@@UNIQUE\_CHECKS**,** UNIQUE\_CHECKS**=**0**;**

**SET** @OLD\_FOREIGN\_KEY\_CHECKS**=**@@FOREIGN\_KEY\_CHECKS**,** FOREIGN\_KEY\_CHECKS**=**0**;**

**SET** @OLD\_SQL\_MODE**=**@@SQL\_MODE**,** SQL\_MODE**=**'TRADITIONAL,ALLOW\_INVALID\_DATES'**;**

-- -----------------------------------------------------

-- Schema mydb

-- -----------------------------------------------------

-- -----------------------------------------------------

-- Schema financedw

-- -----------------------------------------------------

-- -----------------------------------------------------

-- Schema financedw

-- -----------------------------------------------------

**CREATE** **SCHEMA** **IF** **NOT** **EXISTS** `financedw` **DEFAULT** **CHARACTER** **SET** utf8 **;**

**USE** `financedw` **;**

-- -----------------------------------------------------

-- Table `financedw`.`customer`

-- -----------------------------------------------------

**CREATE** **TABLE** **IF** **NOT** **EXISTS** `financedw`**.**`customer` **(**

`customer\_SK` **INT(**11**)** **NOT** **NULL** AUTO\_INCREMENT**,**

`customerID` **INT(**11**)** **NULL** **DEFAULT** **NULL,**

`customerName` **VARCHAR(**50**)** **NULL** **DEFAULT** **NULL,**

`customerTypeID` **CHAR(**1**)** **NULL** **DEFAULT** **NULL,**

`customerTypeName` **VARCHAR(**20**)** **NULL** **DEFAULT** **NULL,**

`customerAddr1` **VARCHAR(**50**)** **NULL** **DEFAULT** **NULL,**

`customerAddr2` **VARCHAR(**50**)** **NULL** **DEFAULT** **NULL,**

`customerCity` **CHAR(**20**)** **NULL** **DEFAULT** **NULL,**

`customerState` **CHAR(**2**)** **NULL** **DEFAULT** **NULL,**

`customerZip` **CHAR(**5**)** **NULL** **DEFAULT** **NULL,**

**PRIMARY** **KEY** **(**`customer\_SK`**))**

ENGINE **=** InnoDB

AUTO\_INCREMENT **=** 173

**DEFAULT** **CHARACTER** **SET** **=** utf8**;**

-- -----------------------------------------------------

-- Table `financedw`.`date\_dimension`

-- -----------------------------------------------------

**CREATE** **TABLE** **IF** **NOT** **EXISTS** `financedw`**.**`date\_dimension` **(**

`dateKey\_SK` **INT(**11**)** **NOT** **NULL,**

`fullDate` **DATE** **NULL** **DEFAULT** **NULL,**

`dateFormatUSA` **CHAR(**11**)** **NOT** **NULL,**

`day\_of\_Week` **TINYINT(**4**)** **NOT** **NULL,**

`day\_NameOfWeek` **CHAR(**10**)** **NOT** **NULL,**

`day\_Of\_Month` **TINYINT(**4**)** **NOT** **NULL,**

`day\_Of\_Year` **SMALLINT(**6**)** **NOT** **NULL,**

`week\_Of\_Year` **TINYINT(**4**)** **NOT** **NULL,**

`month\_Name` **CHAR(**10**)** **NOT** **NULL,**

`month\_Of\_Year` **TINYINT(**4**)** **NOT** **NULL,**

`calenderQuarter` **TINYINT(**4**)** **NOT** **NULL,**

`calenderYear` **SMALLINT(**6**)** **NOT** **NULL,**

`fiscalWeekofYear` **TINYINT(**4**)** **NOT** **NULL,**

`fiscalMonthofYear` **TINYINT(**4**)** **NOT** **NULL,**

`fiscalQuarter` **TINYINT(**4**)** **NOT** **NULL,**

`fiscalYear` **INT(**11**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**`dateKey\_SK`**))**

ENGINE **=** InnoDB

**DEFAULT** **CHARACTER** **SET** **=** utf8**;**

-- -----------------------------------------------------

-- Table `financedw`.`lookup\_state`

-- -----------------------------------------------------

**CREATE** **TABLE** **IF** **NOT** **EXISTS** `financedw`**.**`lookup\_state` **(**

`**state**` **VARCHAR(**25**)** **NULL** **DEFAULT** **NULL,**

`abbrev` **CHAR(**2**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**`abbrev`**))**

ENGINE **=** InnoDB

**DEFAULT** **CHARACTER** **SET** **=** utf8**;**

-- -----------------------------------------------------

-- Table `financedw`.`pec\_junk`

-- -----------------------------------------------------

**CREATE** **TABLE** **IF** **NOT** **EXISTS** `financedw`**.**`pec\_junk` **(**

`Junk\_SK` **INT(**11**)** **NOT** **NULL** AUTO\_INCREMENT**,**

`orderMethod` **VARCHAR(**10**)** **NULL** **DEFAULT** **NULL,**

`paymentMethod` **VARCHAR(**10**)** **NULL** **DEFAULT** **NULL,**

`shippingOption` **TINYINT(**4**)** **NULL** **DEFAULT** **NULL,**

`shippingMethod` **VARCHAR(**10**)** **NULL** **DEFAULT** **NULL,**

**PRIMARY** **KEY** **(**`Junk\_SK`**))**

ENGINE **=** InnoDB

AUTO\_INCREMENT **=** 38

**DEFAULT** **CHARACTER** **SET** **=** utf8**;**

-- -----------------------------------------------------

-- Table `financedw`.`product`

-- -----------------------------------------------------

**CREATE** **TABLE** **IF** **NOT** **EXISTS** `financedw`**.**`product` **(**

`product\_SK` **INT(**11**)** **NOT** **NULL** AUTO\_INCREMENT**,**

`productID` **INT(**11**)** **NOT** **NULL,**

`productDescription` **VARCHAR(**50**)** **NULL** **DEFAULT** **NULL,**

`productTypeID` **INT(**11**)** **NULL** **DEFAULT** **NULL,**

`productTypeDescription` **VARCHAR(**25**)** **NULL** **DEFAULT** **NULL,**

`BUID` **CHAR(**1**)** **NULL** **DEFAULT** **NULL,**

`BUIDName` **VARCHAR(**25**)** **NULL** **DEFAULT** **NULL,**

`abbrev` **VARCHAR(**10**)** **NULL** **DEFAULT** **NULL,**

`price1` **DECIMAL(**10**,**2**)** **NULL** **DEFAULT** **NULL,**

`price2` **DECIMAL(**10**,**2**)** **NULL** **DEFAULT** **NULL,**

`unitCost` **DECIMAL(**10**,**2**)** **NULL** **DEFAULT** **NULL,**

**PRIMARY** **KEY** **(**`product\_SK`**))**

ENGINE **=** InnoDB

AUTO\_INCREMENT **=** 129

**DEFAULT** **CHARACTER** **SET** **=** utf8**;**

-- -----------------------------------------------------

-- Table `financedw`.`supplier`

-- -----------------------------------------------------

**CREATE** **TABLE** **IF** **NOT** **EXISTS** `financedw`**.**`supplier` **(**

`supplier\_SK` **INT(**11**)** **NOT** **NULL** AUTO\_INCREMENT**,**

`supplierID` **INT(**11**)** **NULL** **DEFAULT** **NULL,**

`supplierName` **VARCHAR(**50**)** **NULL** **DEFAULT** **NULL,**

`supplierAddr1` **VARCHAR(**50**)** **NULL** **DEFAULT** **NULL,**

`supplierAddr2` **VARCHAR(**50**)** **NULL** **DEFAULT** **NULL,**

`supplierCity` **VARCHAR(**20**)** **NULL** **DEFAULT** **NULL,**

`supplierState` **CHAR(**2**)** **NULL** **DEFAULT** **NULL,**

`supplierZip` **CHAR(**5**)** **NULL** **DEFAULT** **NULL,**

**PRIMARY** **KEY** **(**`supplier\_SK`**))**

ENGINE **=** InnoDB

AUTO\_INCREMENT **=** 16

**DEFAULT** **CHARACTER** **SET** **=** utf8**;**

-- -----------------------------------------------------

-- Table `financedw`.`salesfact`

-- -----------------------------------------------------

**CREATE** **TABLE** **IF** **NOT** **EXISTS** `financedw`**.**`salesfact` **(**

`customer\_SK` **INT(**11**)** **NOT** **NULL,**

`junk\_SK` **INT(**11**)** **NOT** **NULL** **DEFAULT** '37'**,**

`product\_SK` **INT(**11**)** **NOT** **NULL,**

`orderDate\_SK` **INT(**11**)** **NOT** **NULL,**

`salesDate\_SK` **INT(**11**)** **NOT** **NULL,**

`supplier\_SK` **INT(**11**)** **NOT** **NULL,**

`invoiceID` **INT(**11**)** **NOT** **NULL,**

`division` **VARCHAR(**6**)** **NOT** **NULL,**

`costPrice` **DECIMAL(**10**,**2**)** **NULL** **DEFAULT** '0.00'**,**

`salePrice` **DECIMAL(**10**,**2**)** **NULL** **DEFAULT** '0.00'**,**

`quantity` **DECIMAL(**8**,**2**)** **NULL** **DEFAULT** '0.00'**,**

`performance` **CHAR(**2**)** **NULL** **DEFAULT** **NULL,**

`discount` **INT(**11**)** **NULL** **DEFAULT** **NULL,**

`shippingCost` **DECIMAL(**10**,**2**)** **NULL** **DEFAULT** **NULL,**

**PRIMARY** **KEY** **(**`customer\_SK`**,** `junk\_SK`**,** `product\_SK`**,** `orderDate\_SK`**,** `supplier\_SK`**,** `salesDate\_SK`**,** `invoiceID`**,** `division`**),**

**INDEX** `Supplier\_Sales\_idx` **(**`supplier\_SK` **ASC),**

**INDEX** `Product\_Sales\_idx` **(**`product\_SK` **ASC),**

**INDEX** `Junk\_Sales\_idx` **(**`junk\_SK` **ASC),**

**INDEX** `Order\_Sales\_idx` **(**`orderDate\_SK` **ASC),**

**INDEX** `Sales\_Sales\_idx` **(**`salesDate\_SK` **ASC),**

**CONSTRAINT** `Customer\_Sales`

**FOREIGN** **KEY** **(**`customer\_SK`**)**

**REFERENCES** `financedw`**.**`customer` **(**`customer\_SK`**)**

**ON** **DELETE** **CASCADE**

**ON** **UPDATE** **CASCADE,**

**CONSTRAINT** `Junk\_Sales`

**FOREIGN** **KEY** **(**`junk\_SK`**)**

**REFERENCES** `financedw`**.**`pec\_junk` **(**`Junk\_SK`**)**

**ON** **DELETE** **NO** ACTION

**ON** **UPDATE** **NO** ACTION**,**

**CONSTRAINT** `Order\_Sales`

**FOREIGN** **KEY** **(**`orderDate\_SK`**)**

**REFERENCES** `financedw`**.**`date\_dimension` **(**`dateKey\_SK`**)**

**ON** **DELETE** **NO** ACTION

**ON** **UPDATE** **NO** ACTION**,**

**CONSTRAINT** `Product\_Sales`

**FOREIGN** **KEY** **(**`product\_SK`**)**

**REFERENCES** `financedw`**.**`product` **(**`product\_SK`**)**

**ON** **DELETE** **NO** ACTION

**ON** **UPDATE** **NO** ACTION**,**

**CONSTRAINT** `Sales\_Sales`

**FOREIGN** **KEY** **(**`salesDate\_SK`**)**

**REFERENCES** `financedw`**.**`date\_dimension` **(**`dateKey\_SK`**)**

**ON** **DELETE** **NO** ACTION

**ON** **UPDATE** **NO** ACTION**,**

**CONSTRAINT** `Supplier\_Sales`

**FOREIGN** **KEY** **(**`supplier\_SK`**)**

**REFERENCES** `financedw`**.**`supplier` **(**`supplier\_SK`**)**

**ON** **DELETE** **CASCADE**

**ON** **UPDATE** **CASCADE)**

ENGINE **=** InnoDB

**DEFAULT** **CHARACTER** **SET** **=** utf8**;**

-- -----------------------------------------------------

-- Table `financedw`.`staging\_table`

-- -----------------------------------------------------

**CREATE** **TABLE** **IF** **NOT** **EXISTS** `financedw`**.**`staging\_table` **(**

`serialNumber` **INT(**11**)** **NOT** **NULL** AUTO\_INCREMENT**,**

`division` **VARCHAR(**6**)** **NULL** **DEFAULT** 'TCP-E'**,**

`customerTypeID` **CHAR(**1**)** **NULL** **DEFAULT** **NULL,**

`customerTypeName` **VARCHAR(**20**)** **NULL** **DEFAULT** **NULL,**

`customerID` **INT(**11**)** **NULL** **DEFAULT** **NULL,**

`customerName` **VARCHAR(**50**)** **NULL** **DEFAULT** **NULL,**

`customerAddr1` **VARCHAR(**50**)** **NULL** **DEFAULT** **NULL,**

`customerAddr2` **VARCHAR(**50**)** **NULL** **DEFAULT** **NULL,**

`customerCity` **VARCHAR(**20**)** **NULL** **DEFAULT** **NULL,**

`customerState` **CHAR(**2**)** **NULL** **DEFAULT** **NULL,**

`customerZip` **CHAR(**5**)** **NULL** **DEFAULT** **NULL,**

`invoiceID` **INT(**11**)** **NULL** **DEFAULT** **NULL,**

`salesDate` **DATE** **NULL** **DEFAULT** **NULL,**

`amount` **DECIMAL(**10**,**2**)** **NULL** **DEFAULT** **NULL,**

`quantity` **DECIMAL(**8**,**2**)** **NULL** **DEFAULT** **NULL,**

`productID` **INT(**11**)** **NULL** **DEFAULT** **NULL,**

`productDescription` **VARCHAR(**50**)** **NULL** **DEFAULT** **NULL,**

`productPrice1` **DECIMAL(**10**,**2**)** **NULL** **DEFAULT** **NULL,**

`productPrice2` **DECIMAL(**10**,**2**)** **NULL** **DEFAULT** **NULL,**

`productUnitCost` **DECIMAL(**10**,**2**)** **NULL** **DEFAULT** **NULL,**

`supplierID` **INT(**11**)** **NULL** **DEFAULT** **NULL,**

`supplierName` **VARCHAR(**50**)** **NULL** **DEFAULT** **NULL,**

`supplierAddr1` **VARCHAR(**50**)** **NULL** **DEFAULT** **NULL,**

`supplierAddr2` **VARCHAR(**50**)** **NULL** **DEFAULT** **NULL,**

`supplierCity` **VARCHAR(**20**)** **NULL** **DEFAULT** **NULL,**

`supplierState` **CHAR(**2**)** **NULL** **DEFAULT** **NULL,**

`supplierZip` **CHAR(**5**)** **NULL** **DEFAULT** **NULL,**

`productTypeID` **INT(**11**)** **NULL** **DEFAULT** **NULL,**

`productTypeDescription` **VARCHAR(**25**)** **NULL** **DEFAULT** **NULL,**

`BUID` **CHAR(**1**)** **NULL** **DEFAULT** **NULL,**

`BUIDName` **VARCHAR(**25**)** **NULL** **DEFAULT** **NULL,**

`abbrev` **VARCHAR(**10**)** **NULL** **DEFAULT** **NULL,**

**PRIMARY** **KEY** **(**`serialNumber`**))**

ENGINE **=** InnoDB

AUTO\_INCREMENT **=** 131071

**DEFAULT** **CHARACTER** **SET** **=** utf8**;**

**SET** SQL\_MODE**=**@OLD\_SQL\_MODE**;**

**SET** FOREIGN\_KEY\_CHECKS**=**@OLD\_FOREIGN\_KEY\_CHECKS**;**

**SET** UNIQUE\_CHECKS**=**@OLD\_UNIQUE\_CHECKS**;**

# Data Staging Activities – ETL

## Data Cleansing

|  |  |  |  |
| --- | --- | --- | --- |
| **DM Table** | **Attribute** | **Problem** | https://lh3.googleusercontent.com/JIlzNyrj4_Dv3kN9P3kRfEpm8dIH12tFfSrlFcHKEAsPHxFfwI65wVp1KRc31uqpwa8KCbqOYy5R56SeTqzJfdr0TIHIA5tK5GtguMUJGDCH3HWqsbUJUE6OViUfv8CbWM8FWLqfNeVQDMGtfwhttps://lh4.googleusercontent.com/cYoKX_e_ISMqwl2o1rS6YKFbnbosRyHThSe7spFrsQbmpiYafjQFCJ5gXYsDleIzcjFS_KhPcIGCDl_Cly9IF4IhUg9V8pQyHmdDWQg5_-Kw7FoMmfTcNAIaZTrC78-WF-8VZT6L6LIbpV-XJg  **Resolution Strategy**  (attach code) |
| TPCWcustomer.csv | state | Needs to be two letter code | In Pentaho, Used “Database lookup” to change all state to two letter code |
| TPCWcustomer.csv | zip | One zipcode is only 4 digits | Removed it |
| TPCWcustomer.csvhttps://lh3.googleusercontent.com/JIlzNyrj4_Dv3kN9P3kRfEpm8dIH12tFfSrlFcHKEAsPHxFfwI65wVp1KRc31uqpwa8KCbqOYy5R56SeTqzJfdr0TIHIA5tK5GtguMUJGDCH3HWqsbUJUE6OViUfv8CbWM8FWLqfNeVQDMGtfw | custtype | It doesn’t match the required format in custtype file (Comm should be Commercial, Edu should be Education) and so on | In Pentaho, Used “Replace In String” to change the fields in the required format |
| TPCWproduct.csv | Attn - Field #7 | Column heading given as ‘Attn’ | Renamed to ‘Addr 1’ |
| TPCWproduct.csv | Product type ID | - Double quotes exist  - ID has 0s preceding the value.eg. ‘0001’ format | Removed double quotes using Regex  ID formatted to ‘1’ using |
| TPCWproduct.csv | Product ID | - Double quotes exist  - ID given in ‘0001’ format | Removed double quotes using Regex  ID formatted to ‘1’ |
| https://lh3.googleusercontent.com/JIlzNyrj4_Dv3kN9P3kRfEpm8dIH12tFfSrlFcHKEAsPHxFfwI65wVp1KRc31uqpwa8KCbqOYy5R56SeTqzJfdr0TIHIA5tK5GtguMUJGDCH3HWqsbUJUE6OViUfv8CbWM8FWLqfNeVQDMGtfwTPCWsales\_new.csv | custID | There is one negative value | Removed the negative sign |
| https://lh3.googleusercontent.com/JIlzNyrj4_Dv3kN9P3kRfEpm8dIH12tFfSrlFcHKEAsPHxFfwI65wVp1KRc31uqpwa8KCbqOYy5R56SeTqzJfdr0TIHIA5tK5GtguMUJGDCH3HWqsbUJUE6OViUfv8CbWM8FWLqfNeVQDMGtfwTPCWsales\_new.csv | custID | There is null value with invoice number 19061 | Removed it |
| https://lh3.googleusercontent.com/JIlzNyrj4_Dv3kN9P3kRfEpm8dIH12tFfSrlFcHKEAsPHxFfwI65wVp1KRc31uqpwa8KCbqOYy5R56SeTqzJfdr0TIHIA5tK5GtguMUJGDCH3HWqsbUJUE6OViUfv8CbWM8FWLqfNeVQDMGtfwTPCWsales\_new.csv | entire row | There are null values with invoice number 26511 | Removed the entire row |
| https://lh3.googleusercontent.com/JIlzNyrj4_Dv3kN9P3kRfEpm8dIH12tFfSrlFcHKEAsPHxFfwI65wVp1KRc31uqpwa8KCbqOYy5R56SeTqzJfdr0TIHIA5tK5GtguMUJGDCH3HWqsbUJUE6OViUfv8CbWM8FWLqfNeVQDMGtfwTPCWsales\_new.csv | qty | There is null value with invoice number 3032 | Replaced it with 0 |
| https://lh3.googleusercontent.com/JIlzNyrj4_Dv3kN9P3kRfEpm8dIH12tFfSrlFcHKEAsPHxFfwI65wVp1KRc31uqpwa8KCbqOYy5R56SeTqzJfdr0TIHIA5tK5GtguMUJGDCH3HWqsbUJUE6OViUfv8CbWM8FWLqfNeVQDMGtfwTPCWsales\_new.csv | discounted | There is null value with invoice number 45461 | Put default value 0 |
| https://lh3.googleusercontent.com/JIlzNyrj4_Dv3kN9P3kRfEpm8dIH12tFfSrlFcHKEAsPHxFfwI65wVp1KRc31uqpwa8KCbqOYy5R56SeTqzJfdr0TIHIA5tK5GtguMUJGDCH3HWqsbUJUE6OViUfv8CbWM8FWLqfNeVQDMGtfwTPCWsales\_new.csv | salesDate | The date format is different. | Formatted to mm/dd/yyyy format using pentaho transformation |
| PECcustomer.csv | address | Dirty characters in address field | Dirty characters are removed |
| PECcustomer.csv | custtype | doesn’t follow the format in custtype file | In Pentaho, Using “Replace In String” to change STATELOCALGOVT into  State/Local Gov and usgovt into US Govt |
| https://lh3.googleusercontent.com/JIlzNyrj4_Dv3kN9P3kRfEpm8dIH12tFfSrlFcHKEAsPHxFfwI65wVp1KRc31uqpwa8KCbqOYy5R56SeTqzJfdr0TIHIA5tK5GtguMUJGDCH3HWqsbUJUE6OViUfv8CbWM8FWLqfNeVQDMGtfwPECsales\_new.csv | Entire row | There is a problem in data format in row number 41002 | Removed the row |
| https://lh3.googleusercontent.com/JIlzNyrj4_Dv3kN9P3kRfEpm8dIH12tFfSrlFcHKEAsPHxFfwI65wVp1KRc31uqpwa8KCbqOYy5R56SeTqzJfdr0TIHIA5tK5GtguMUJGDCH3HWqsbUJUE6OViUfv8CbWM8FWLqfNeVQDMGtfwPECsales\_new.csv | salesDate | There is a problem in date format in record number 72 and 20000 | Fixed it |
| PECsales\_new.csv | orderMethod | OrderMethod has a Date value | Removed the row |
| PECsales\_new.csv | shipMethod | shipMethod has a number value | Removed the row |
| PECproduct.csv | supplierName | supplierName is null | update supplierName to PEC where null and removed double quotes using Regex |
| PECproduct.csv | Unit cost | There are some null values. | Calculated manufacturing cost / qty |
| PECproduct.csv | Product type ID | Double quotes exist in the data | Removed double quotes using Regex |
| PECproduct.csv | Product ID | Double quotes exist in the data | Removed double quotes using Regex |
| PECproduct.csv | Product Description | Double quotes exist in the data | Removed double quotes using Regex |
| ManufacturingCost.csv | All the rows | Separated by pipes | changed pipes to semi colon |
| ManufacturingCost.csv | Year | Represented as Y in a single digit. | Transformed to YYYY format adding 2000 to the year column |

## Data Transformation

|  |  |
| --- | --- |
| **DM Table** | **Image Creation Process** (attach code) |
| TPC-W Customer | Customer Dimension with SK  Creation of TPC-W Customer Dimension  Customer Dimension Lookup |
| TPC-W Product | Product Dimension with SK  Product and Supplier Cleaning |
| TPC-W Supplier | Supplier Dimension creation |
| TPC-W Date | Date Cleaning  **Stored Procedure code for date\_dimension**  delimiter **//**  **drop** **procedure** **if** **exists** PopulateDateDimension**//**  **CREATE** **PROCEDURE** PopulateDateDimension**(**BeginDate **DATETIME,** EndDate **DATETIME)**  **BEGIN**  # **Number** **of** months **to** **add** **to** the **date** **to** **get** the **current** Fiscal **date**  **DECLARE** FiscalYearMonthsOffset **INT;**  # These two counters **are** used **in** our **loop.**  **DECLARE** DateCounter **DATETIME;** #**Current** **date** **in** **loop**  **DECLARE** FiscalCounter **DATETIME;** #Fiscal **Year** **Date** **in** **loop**  # **Set** this **to** the **number** **of** months **to** **add** **to** the **current** **date** **to** **get**  # the beginning **of** the Fiscal **year.** **For** example**,** **if** the Fiscal **year**  # begins July 1**,** put a 6 there**.**  # Negative **values** **are** also allowed**,** thus **if** your 2010 Fiscal **year**  # begins **in** July **of** 2009**,** put a -6.  **SET** FiscalYearMonthsOffset **=** -3**;**  # **Start** the counter **at** the **begin** **date**  **SET** DateCounter **=** BeginDate**;**  **WHILE** DateCounter **<=** EndDate **DO**  # Calculate the **current** Fiscal **date** **as** an offset **of**  # the **current** **date** **in** the **loop**  **SET** FiscalCounter **=** **DATE\_ADD(**DateCounter**,** **INTERVAL** FiscalYearMonthsOffset **MONTH);**  # **add** a **record** **into** the **date** dimension **table** **for** this **date**  **INSERT** **INTO** Date\_Dimension  **(**dateKey\_SK**,**  fullDate**,**  dateFormatUSA**,**  day\_Of\_Week**,**  day\_NameOfWeek**,**  day\_Of\_Month**,**  day\_Of\_Year**,**  week\_Of\_Year**,**  month\_Name**,**  month\_Of\_Year**,**  calenderQuarter**,**  calenderYear**,**  fiscalWeekofYear**,**  fiscalMonthofYear**,**  fiscalQuarter**,**  fiscalYear**)**  **VALUES** **(**  **(** **YEAR(**DateCounter**)** **\*** 10000 **)** **+** **(** **MONTH(**DateCounter**)\*** 100 **)** **+** **DAY(**DateCounter**),** #DateKey  DateCounter**,** # FullDate  **CONCAT(DATE\_FORMAT(**DateCounter**,**'%m'**),**'/'**,DATE\_FORMAT(**DateCounter**,**'%d'**),**'/'**,CAST(YEAR(**DateCounter**)** **AS** **CHAR(**4**))),**#DateNameUS  DAYOFWEEK**(**DateCounter**),** #DayOfWeek  DAYNAME**(**DateCounter**),** #DayNameOfWeek  DAYOFMONTH**(**DateCounter**),** #DayOfMonth  DAYOFYEAR**(**DateCounter**),** #DayOfYear  WEEKOFYEAR**(**DateCounter**),** #WeekOfYear  MONTHNAME**(**DateCounter**),** #MonthName  **MONTH(**DateCounter**),** #MonthOfYear  QUARTER**(**DateCounter**),** #CalendarQuarter  **YEAR(**DateCounter**),** #CalendarYear  WEEKOFYEAR**(**FiscalCounter**),** #FiscalWeekOfYear  **MONTH(**FiscalCounter**),** #FiscalMonthOfYear  QUARTER**(**FiscalCounter**),** #FiscalQuarter  **YEAR(**FiscalCounter**),** #FiscalYear  **);**  # **Increment** the **date** counter **for** **next** pass thru the **loop**  **SET** DateCounter **=** **DATE\_ADD(**DateCounter**,** **INTERVAL** 1 **DAY);**  **END** **WHILE;**  **END//** |
| TPC-W Sales | Sale Date and Junk Lookup    Salefact Creation |
| PEC- Junk | **Code to generate the junk dimension manually:**  -- Table structure for table `pec\_junk`  **DROP** **TABLE** **IF** **EXISTS** `pec\_junk`**;**  **CREATE** **TABLE** `pec\_junk` **(**  `Junk\_SK` **int(**11**)** **NOT** **NULL** AUTO\_INCREMENT**,**  `orderMethod` **varchar(**10**)** **DEFAULT** **NULL,**  `paymentMethod` **varchar(**10**)** **DEFAULT** **NULL,**  `shippingOption` **tinyint(**4**)** **DEFAULT** **NULL,**  `shippingMethod` **varchar(**10**)** **DEFAULT** **NULL,**  **PRIMARY** **KEY** **(**`Junk\_SK`**)**  **)** ENGINE**=**InnoDB AUTO\_INCREMENT**=**38 **DEFAULT** CHARSET**=**utf8**;**  --  -- Dumping data for table `pec\_junk`  --  **LOCK** TABLES `pec\_junk` **WRITE;**  /\*!40000 ALTER TABLE `pec\_junk` DISABLE KEYS \*/**;**  **INSERT** **INTO** `pec\_junk` **VALUES** **(**1**,**'Internet'**,**'COD'**,**0**,**'N/A'**),(**2**,**'Phone'**,**'COD'**,**0**,**'N/A'**),(**3**,**'Email'**,**'COD'**,**0**,**'N/A'**),(**4**,**'Internet'**,**'Charge'**,**0**,**'N/A'**),(**5**,**'Phone'**,**'Charge'**,**0**,**'N/A'**),(**6**,**'Email'**,**'Charge'**,**0**,**'N/A'**),(**7**,**'Internet'**,**'Cash'**,**0**,**'N/A'**),(**8**,**'Phone'**,**'Cash'**,**0**,**'N/A'**),(**9**,**'Email'**,**'Cash'**,**0**,**'N/A'**),(**10**,**'Internet'**,**'COD'**,**1**,**'Train'**),(**11**,**'Phone'**,**'COD'**,**1**,**'Train'**),(**12**,**'Email'**,**'COD'**,**1**,**'Train'**),(**13**,**'Internet'**,**'Charge'**,**1**,**'Train'**),(**14**,**'Phone'**,**'Charge'**,**1**,**'Train'**),(**15**,**'Email'**,**'Charge'**,**1**,**'Train'**),(**16**,**'Internet'**,**'Cash'**,**1**,**'Train'**),(**17**,**'Phone'**,**'Cash'**,**1**,**'Train'**),(**18**,**'Email'**,**'Cash'**,**1**,**'Train'**),(**19**,**'Internet'**,**'COD'**,**2**,**'Truck'**),(**20**,**'Phone'**,**'COD'**,**2**,**'Truck'**),(**21**,**'Email'**,**'COD'**,**2**,**'Truck'**),(**22**,**'Internet'**,**'Charge'**,**2**,**'Truck'**),(**23**,**'Phone'**,**'Charge'**,**2**,**'Truck'**),(**24**,**'Email'**,**'Charge'**,**2**,**'Truck'**),(**25**,**'Internet'**,**'Cash'**,**2**,**'Truck'**),(**26**,**'Phone'**,**'Cash'**,**2**,**'Truck'**),(**27**,**'Email'**,**'Cash'**,**2**,**'Truck'**),(**28**,**'Internet'**,**'COD'**,**3**,**'Air'**),(**29**,**'Phone'**,**'COD'**,**3**,**'Air'**),(**30**,**'Email'**,**'COD'**,**3**,**'Air'**),(**31**,**'Internet'**,**'Charge'**,**3**,**'Air'**),(**32**,**'Phone'**,**'Charge'**,**3**,**'Air'**),(**33**,**'Email'**,**'Charge'**,**3**,**'Air'**),(**34**,**'Internet'**,**'Cash'**,**3**,**'Air'**),(**35**,**'Phone'**,**'Cash'**,**3**,**'Air'**),(**36**,**'Email'**,**'Cash'**,**3**,**'Air'**),(**37**,NULL,NULL,NULL,NULL);**  /\*!40000 ALTER TABLE `pec\_junk` ENABLE KEYS \*/**;**  UNLOCK TABLES**;** |
| PEC- Customer | Customer cleaning    Customer dimension check  Customer dimension lookup |
| PEC- Manufacturing Cost |  |
| PEC-Unit Cost |  |
| PEC-Product | Product Cleaning      Product Dimension  Product with Unit Cost |
| PEC-Supplier | Supplier Dimension |
| PEC-Date | Date Cleaning  **Stored Procedure code for date\_dimension**  delimiter **//**  **drop** **procedure** **if** **exists** PopulateDateDimension**//**  **CREATE** **PROCEDURE** PopulateDateDimension**(**BeginDate **DATETIME,** EndDate **DATETIME)**  **BEGIN**  # **Number** **of** months **to** **add** **to** the **date** **to** **get** the **current** Fiscal **date**  **DECLARE** FiscalYearMonthsOffset **INT;**  # These two counters **are** used **in** our **loop.**  **DECLARE** DateCounter **DATETIME;** #**Current** **date** **in** **loop**  **DECLARE** FiscalCounter **DATETIME;** #Fiscal **Year** **Date** **in** **loop**  # **Set** this **to** the **number** **of** months **to** **add** **to** the **current** **date** **to** **get**  # the beginning **of** the Fiscal **year.** **For** example**,** **if** the Fiscal **year**  # begins July 1**,** put a 6 there**.**  # Negative **values** **are** also allowed**,** thus **if** your 2010 Fiscal **year**  # begins **in** July **of** 2009**,** put a -6.  **SET** FiscalYearMonthsOffset **=** -3**;**  # **Start** the counter **at** the **begin** **date**  **SET** DateCounter **=** BeginDate**;**  **WHILE** DateCounter **<=** EndDate **DO**  # Calculate the **current** Fiscal **date** **as** an offset **of**  # the **current** **date** **in** the **loop**  **SET** FiscalCounter **=** **DATE\_ADD(**DateCounter**,** **INTERVAL** FiscalYearMonthsOffset **MONTH);**  # **add** a **record** **into** the **date** dimension **table** **for** this **date**  **INSERT** **INTO** Date\_Dimension  **(**dateKey\_SK**,**  fullDate**,**  dateFormatUSA**,**  day\_Of\_Week**,**  day\_NameOfWeek**,**  day\_Of\_Month**,**  day\_Of\_Year**,**  week\_Of\_Year**,**  month\_Name**,**  month\_Of\_Year**,**  calenderQuarter**,**  calenderYear**,**  fiscalWeekofYear**,**  fiscalMonthofYear**,**  fiscalQuarter**,**  fiscalYear**)**  **VALUES** **(**  **(** **YEAR(**DateCounter**)** **\*** 10000 **)** **+** **(** **MONTH(**DateCounter**)\*** 100 **)** **+** **DAY(**DateCounter**),** #DateKey  DateCounter**,** # FullDate  **CONCAT(DATE\_FORMAT(**DateCounter**,**'%m'**),**'/'**,DATE\_FORMAT(**DateCounter**,**'%d'**),**'/'**,CAST(YEAR(**DateCounter**)** **AS** **CHAR(**4**))),**#DateNameUS  DAYOFWEEK**(**DateCounter**),** #DayOfWeek  DAYNAME**(**DateCounter**),** #DayNameOfWeek  DAYOFMONTH**(**DateCounter**),** #DayOfMonth  DAYOFYEAR**(**DateCounter**),** #DayOfYear  WEEKOFYEAR**(**DateCounter**),** #WeekOfYear  MONTHNAME**(**DateCounter**),** #MonthName  **MONTH(**DateCounter**),** #MonthOfYear  QUARTER**(**DateCounter**),** #CalendarQuarter  **YEAR(**DateCounter**),** #CalendarYear  WEEKOFYEAR**(**FiscalCounter**),** #FiscalWeekOfYear  **MONTH(**FiscalCounter**),** #FiscalMonthOfYear  QUARTER**(**FiscalCounter**),** #FiscalQuarter  **YEAR(**FiscalCounter**),** #FiscalYear  **);**  # **Increment** the **date** counter **for** **next** pass thru the **loop**  **SET** DateCounter **=** **DATE\_ADD(**DateCounter**,** **INTERVAL** 1 **DAY);**  **END** **WHILE;**  **END//** |
| PEC-Performance(No of days b/w saleDate and orderDate |  |
| PEC-Salesfact | Date and PEC lookup    Salefact creation |

## Table Population

|  |  |
| --- | --- |
| **DM Table** | **Table Population Process** (attach code) |
| TPC-W Customer |  |
| TPC-W Product |  |
| TPC-W Supplier |  |
| TPC-W Fact | https://lh6.googleusercontent.com/c3iBO4xE__n4HY-2TSlP-Y2CdrJos46SVbe_4yN9VucWiUq6bA3K9fbdhXu88mntUn_VChqR0etiHlH7a-eUilQ5kZBLaV-JvU1ku0defK1l2_AwpDBiaz4oQczI_6gfAYtsNYxC |
| PEC-Customer |  |
| PEC- Product |  |
| PEC- Supplier |  |
| PEC- Fact | https://lh6.googleusercontent.com/o23V0Pa_o8wRrezU4UC5mZcYmDH6XtA0gZ9WkBqFFNZRWcFIT2adpGw4R2UHfMQIR9ZyYuEhKvC_D_BmHBThyEe9dSJEYdlsyOK2xOMtDTJQLOL0CaqHH3voHptcsG-e4eWst1jj |
| TPC-W  Job |  |
| PEC-  Job |  |

# End User Applications

## Queries

|  |
| --- |
| **User Question/Reporting Need** |
| The average time in days needed to fulfill an order from PEC. |
| **SQL Code** |
| **SELECT** **AVG(**performance**)**  **FROM** salesfact  **WHERE** division **=** "PEC"**;** |
| **Supporting Index(es)** |
| INDEX.png |

|  |
| --- |
| **User Question/Reporting Need** |
| The average cost of shipping for a particular product by different methods |
| **SQL Code** |
| **SELECT** p**.**productDescription**,** s**.**shippingMethod**,** **AVG(**s**.**shippingCost**)**  **FROM** salesfact s  **INNER** **JOIN** product **ON** s**.**product\_SK**=**p**.**product\_SK  **INNER** **JOIN** pec\_junk **ON** s**.**Junk\_SK**=**j**.**Junk\_SK  **WHERE** s**.**product\_SK **=** 152  **GROUP** **BY** s**.**shippingMethod**;** |
| **Supporting Index(es)** |
| There is no need index because product\_SK is indexed by  default |

|  |
| --- |
| **User Question/Reporting Need** |
| The most frequent method of ordering a product from PEC. |
| **SQL Code** |
| **SELECT** j**.**orderMethod**,COUNT(**j**.**orderMethod**)**  **FROM** pec\_junk j **INNER** **JOIN** salesfact s **ON** j**.**Junk\_SK **=** s**.**Junk\_SK  **WHERE** s**.**division **=** "PEC"  **GROUP** **BY** j**.**orderMethod**;** |
| **Supporting Index(es)** |
| INDEX.png |

## Views & Summaries

The purpose of this view is to create a top customer report, which will help the company to easily find its top customer information.

**CREATE** **VIEW** TopCustomer **AS**

**SELECT** c**.**customerName**,** c**.**customerCity**,** c**.**customerState**,** c**.**customerTypeName**,** **SUM(**s**.**salePrice**)**

**FROM** salesfact s **INNER** **JOIN** customer c **ON** s**.**customer\_SK **=** c**.**customer\_SK

**GROUP** **BY** c.customerName**;**

# Handling Slowly Changing Dimensions (SCD)

We chose customer dimension table to implement Slowly Changing Dimension.

customerID (**Type0 No Change**)

custType, (**Type 1 Overwrite**)

customerName ,custAddr1,custAddr2,custCity,custState,custZip (**Type 2 Creating a new additional record. All history of dimension changes is kept in the database**)

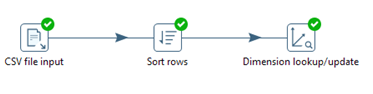
**Type1**: It is necessary when data error happened. The old dimension value is simply overwritten be the new one. For instance, change custType from Education to State/Local Gov.

**Type2**:  Creating a new additional record. All history of dimension changes is kept in the database. You capture attribute change by adding a new row with a new surrogate key to the dimension table. Both the prior and new rows contain as attributes the natural key.

**Implementation**

We used Dimensional lookup/update in Pentaho to implement Type1.

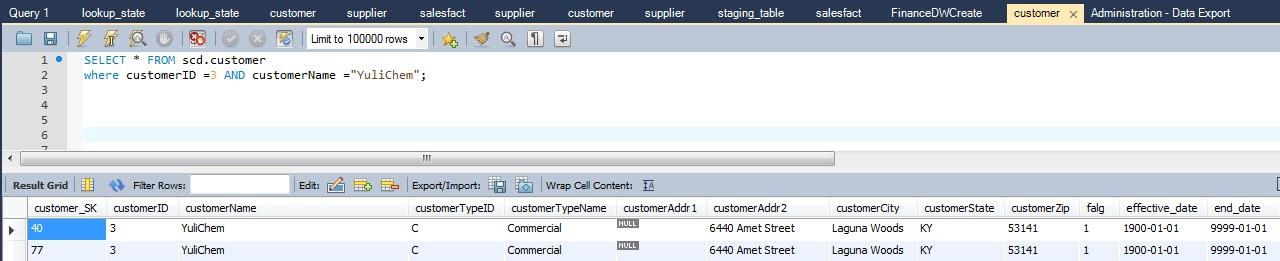
**SCD TYPE 1:**



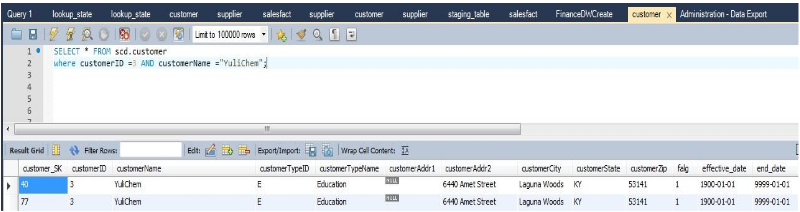
**Sample 1:**

https://lh4.googleusercontent.com/lIfhcDXyBIMDKHIb3Kx1zsIIk0nwQxqL9gMqCbiA8OE6mNKgcvuJA711fnPxRBdejRpjDk9CcQX78szSN2VpWGq7E2paKld0QoojeMxo_GGqDSc2V3FCzb15smdl1p-oW2V1t5Cjj8Jlkwz1LQ

**Original row**



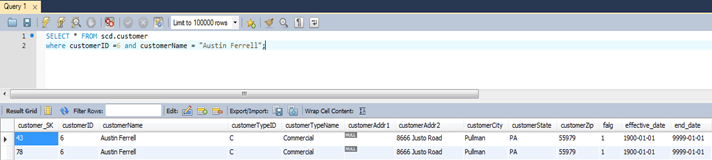
**After run transformation “scd\_type1.ktr”**

****

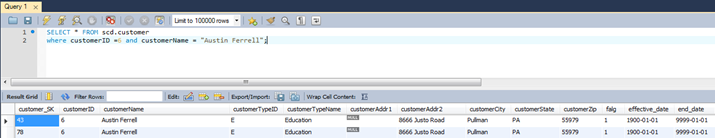
**Sample 2**



**Original row**



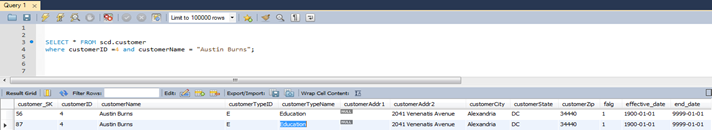
**After run transformation “scd\_type1.ktr”**



**Sample 3**



**Original row**



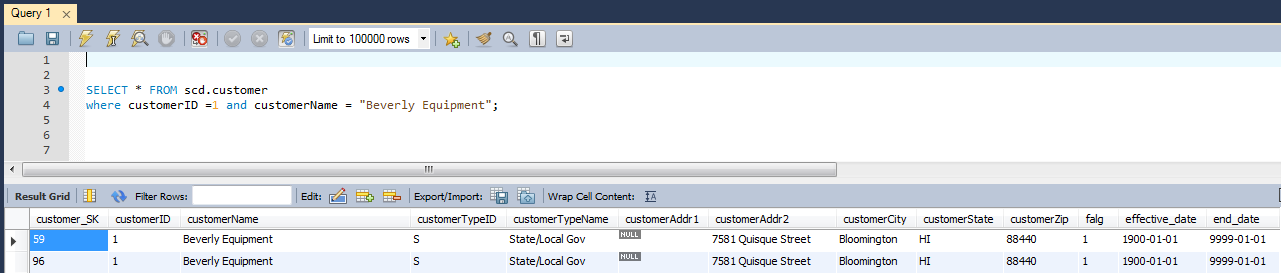
**After run transformation**



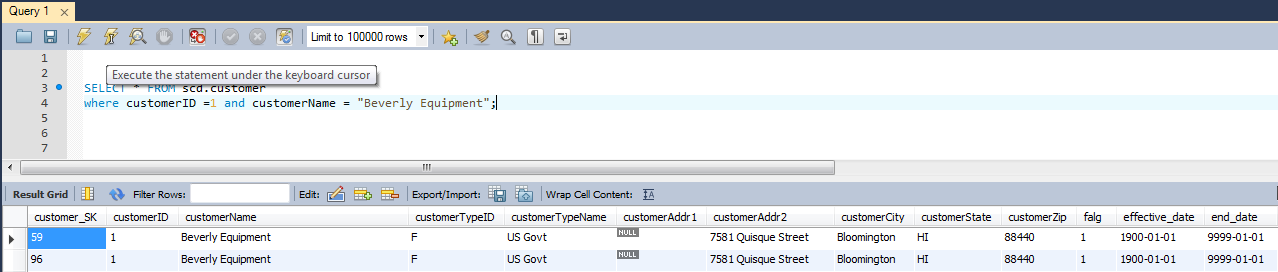
**Sample 4**



**Original row**



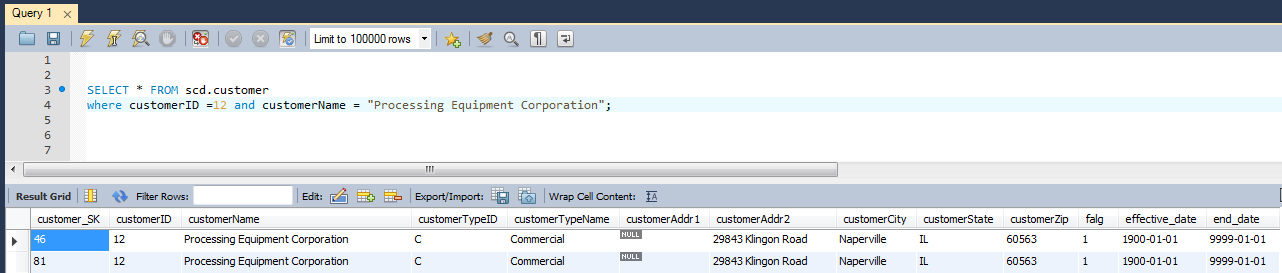
**After run transformation**



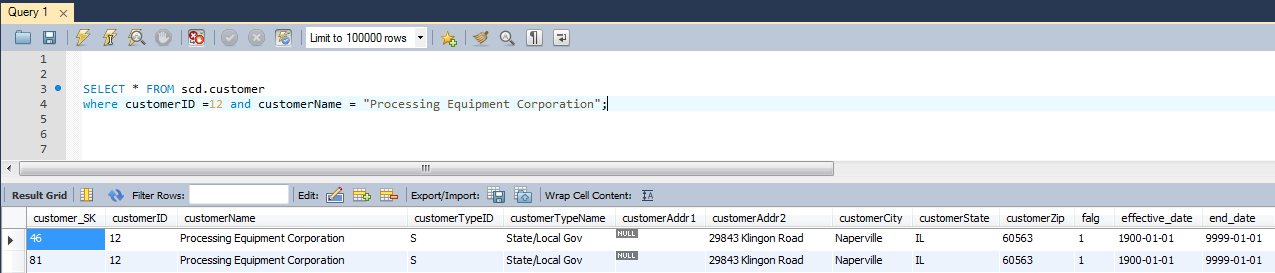
**Sample 5**



**Original 5**



**After run transformation**



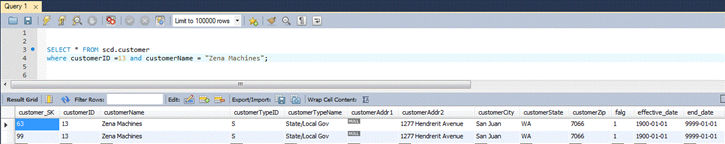
**SCD TYPE 2**



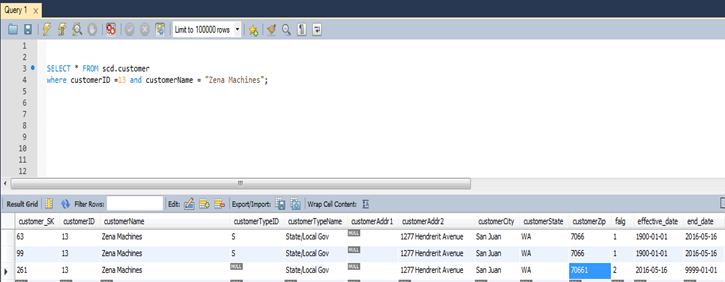
**Sample 1**



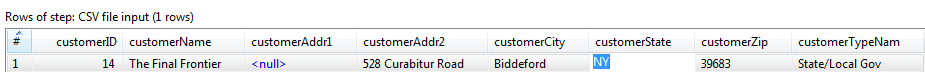
**Original**



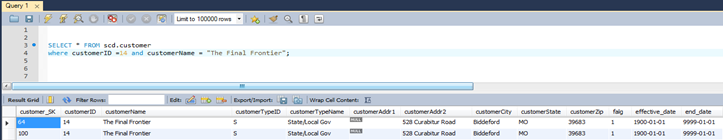
**After run transformation scd\_type2.ktr**



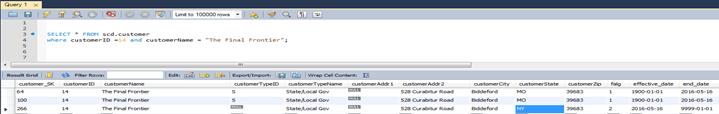
**Sample 2**



**Original row**



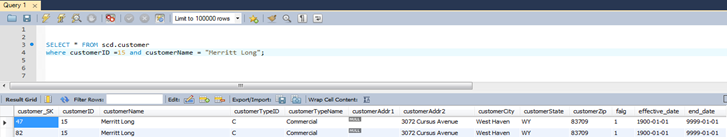
**After Transformation**



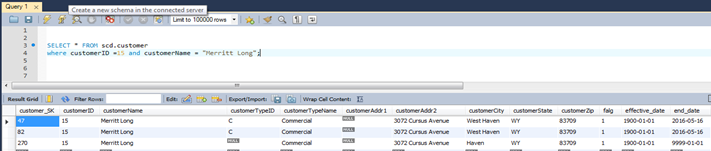
**Sample 3**



**Original row**



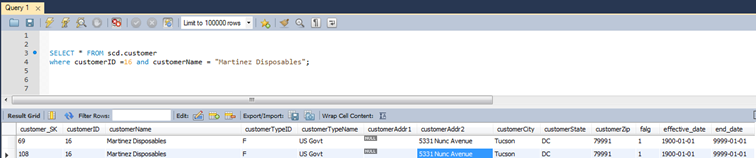
**After transformation**



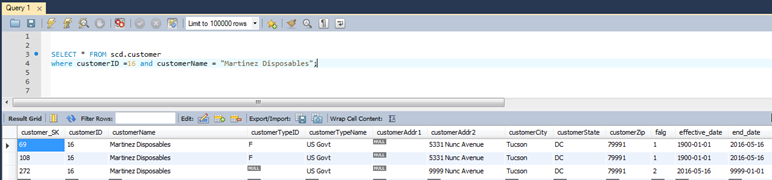
**Sample 4**



**Original row**

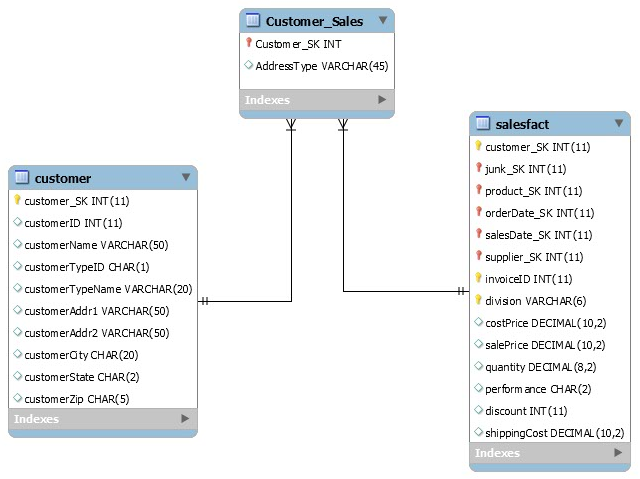


**After Transformation**



# Many-to-Many (N-M) Relationship Implementation Option

There are some strategies to implement many-to-many relationships for data warehousing. One of them is shown below.



* **Bridge Table Method Strateg**y, a bridge table will be used between the two dimensions or between a dimension and the fact table. This table will handle many to many relationship between the two tables. In bridge table we have the surrogate keys for the two tables in addition to a weighting factor column (Song & Rowen, 2001).

**Example:** Customer and salesfact can have a bridging table called Customer\_Sales to list additional information say addresstype (home or work) relating to the customer for a particular sale.

* **Multiple Column Method Strategy**, we should have a number of columns according to the choices that are available to the user. This approach is useful when we have a set of different choices or categories that related to defined record.
* **Boolean Column Method Strategy**, we add a column in the dimension table for each possible value that we may have, where each column will be either single integer as (0 or 1) or a single character as (Y or N) (Hamel, 2007). Usually this method is used when you have a predefined possible values.

# References

* Hammel, R. (2007, January 19). Implementing Many-to-many Relationships in Data Warehousing. Retrieved May 16, 2016, from https://www.pythian.com/blog/implementing-many-to-many-relationships-in-data-warehousing/
* Rowen, W., Song, I. Y., Medsker, C., & Ewen, E. (2001). An analysis of many-to-many relationships between fact and dimension tables in dimensional modeling. In *Proceedings of the International Workshop on Design and Management of Data Warehouses (DMDW 2001), Interlaken, Switzerland*.