**ISTE 724 - Data Warehousing**

**Lab 3 – Report**

***Team 3***

***Ravikiran Jois Yedur Prabhakar (ry4287)***

***Shaivya Chandra (sc2914)***

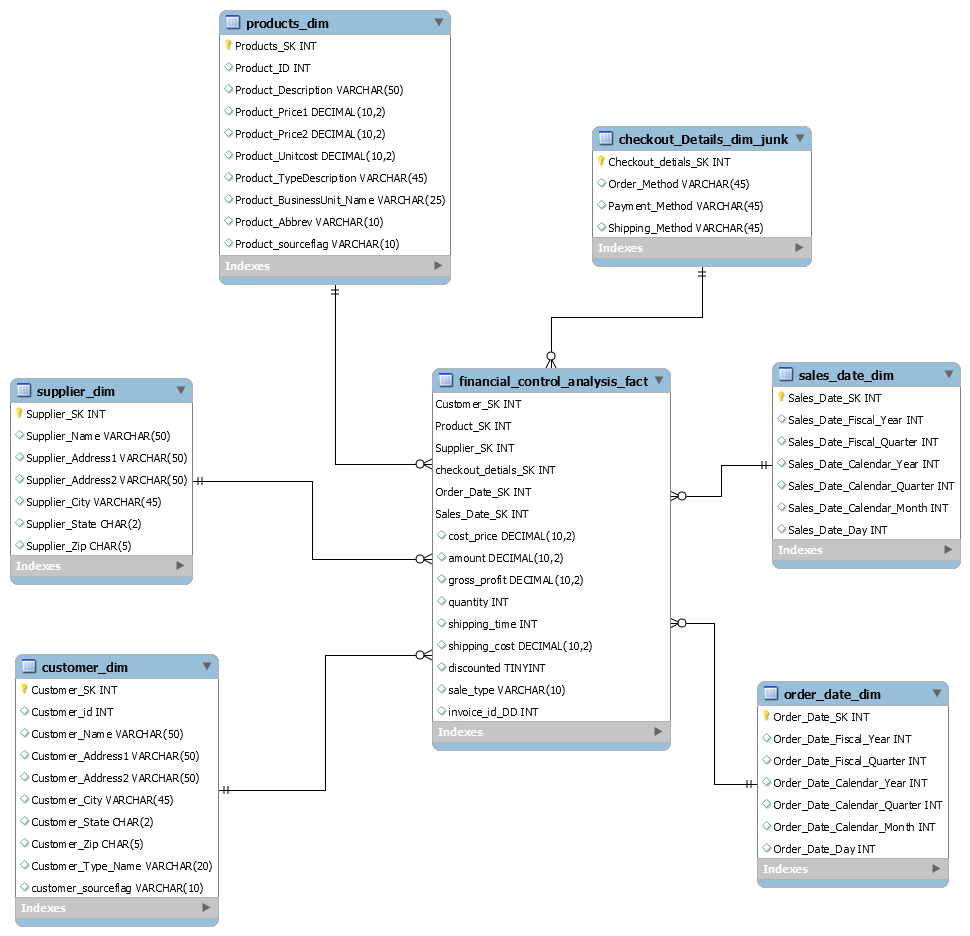
***Suhas Cholleti (sc3614)***

**Introduction**

In this Lab, we aimed to take the previously created star schema, clean and transform the data and load the data. We were supposed to use various applications like SAS, Pentaho PDI, and MySQL Workbench to complete the entire assignment. In the end, we also used the data warehouse to get various reports generated using the SQL Queries.

**Finalize Data Mart Dimensional Model**

Our first step was to decide on the model during which we found that our model was similar. For differentiating between the TPCW and PEC invoices we added the Sale\_Type in the Fact Table. Apart from that, we calculated aggregations like Cost price for the invoice, Gross Profit, Shipping Time, and Shipping cost using Pentaho PDI.



**Data Cleaning**

We found several discrepancies on first looking at the CSV files. We found that the CSV file contained quotes, inconsistent delimiter in data (leading to shifting columns), irregular data, and missing data. We aimed to automate the entire process of cleaning and not use any hardcoded values throughout the process. We also used Pentaho (the ) files to transform the files for the missing values and incorrect values. The below table presents the cleaning steps taken in each file along with the problem and our resolution strategies:

***Data Cleansing Table***

|  |  |  |  |
| --- | --- | --- | --- |
| **File (.csv)** | **Attribute** | **Problem** | **Resolution Strategy (With Attached Code)** |
| **TPCWproduct\_type.csv** | All attributes | Extra spaces and quotes | Removed extra spaces and quotes using SAS code **(Combined\_Code.sas)** |
| typeDescription | Inconsistent abbreviations | Used the tranwrd method in SAS to replace these occurrences **(Combined\_Code.sas)** |
| **TPCWbusiness\_unit.csv** | Name | Inconsistent abbreviations | Used an if condition to insert Misc. as the abbreviation for business unit name Miscellaneous **(Combined\_Code.sas)** |
| **PECbusiness\_unit.csv** | All attributes | Extra spaces and quotes | Removed extra spaces and quotes using SAS code **(Combined\_Code.sas)** |
| name | Inconsistent abbreviations | Used an if condition to insert Misc. as the abbreviation for business unit name Miscellaneous **(Combined\_Code.sas)** |
| All attributes | Inconsistent delimiter | Due to the presence of unwanted semi-colons, the document was not easily readable. We fixed that by splitting the string and inserting the data into the respective columns as required **(Combined\_Code.sas)** |
| **TPCWcustomer\_type.csv** | typeName | Inconsistent abbreviations | Used the tranwrd method in SAS to replace these occurrences (Combined\_Code.sas) |
| **PECcustomer\_type.csv** | All attributes | Extra spaces and quotes | Removed extra spaces and quotes using SAS code **(Combined\_Code.sas)** |
| typename\_original | Inconsistent abbreviations | Used the tranwrd method in SAS to replace these occurrences **(Combined\_Code.sas)** |
| All attributes | Inconsistent delimiter | Due to the presence of unwanted semi-colons, the document was not easily readable. We fixed that by splitting the string and inserting the data into the respective columns as required **(Combined\_Code.sas)** |
| **PECmanufacturingCosts.csv** | year | Year format missing leading 0 | Fixed this using an if the condition that checks if the length of the year is 1 (**Combined\_Code.sas)** |
| **PECproduct\_type.csv** | All attributes | Extra spaces and quotes | Removed extra spaces and quotes using SAS code (**Combined\_Code.sas)** |
| typename\_original | Inconsistent abbreviations | Used the tranwrd method in SAS to replace these occurrences **(Combined\_Code.sas)** |
| All attributes | Inconsistent delimiter | Due to the presence of unwanted semi-colons, the document was not easily readable. We fixed that by splitting the string and inserting the data into the respective columns as required **(Combined\_Code.sas)** |
| **PECcustomer.csv** | custtype, name, address | Inconsistent abbreviations | Used the tranwrd method in SAS to replace these occurrences **(PECcustomer\_Code.sas)** |
| Zip | Zip format missing leading 0 if it is of length 4 | Fixed this using the concatenation operation using || after using strip and scan function to slice the string and concatenate it **(PECcustomer\_Code.sas)** |
| Address | Addr1 and Addr2 split | Split the single attribute into two by using the scan method of SAS using a delimiter **(PECcustomer\_Code.sas)** |
| **PECinvoice.csv** | All attributes | Inconsistent delimiter | Due to the presence of unwanted semi-colons, the document was not easily readable. We fixed that by splitting the string and inserting the data into the respective columns as required **(PECInvoice\_Code.sas)** |
| salesDate | Format inconsistencies | The date field was in the form of both MM-DD-YY or MM/DD/YY. There were also wrongly merged columns with salesDate. We fixed this by splitting the string and then rejoining them to get the data in a uniform format **(PECInvoice\_Code.sas)** |
| shipMethod | Inconsistent values | Using an if condition, the inconsistent values for the shipMethod have been converted to obtain a uniform style **(PECInvoice\_Code.sas)** |
| amt | Incorrect Values | Calculate the new amount using price1, price2 PECproduct\_Transformed.csv and qty, discounted from PECInvoice.csv **(PECinvoice\_Transformation.ktr)** |
| **PECproduct.csv** | All attributes | Extra spaces and quotes | Removed extra spaces and quotes using SAS code **(PECproduct\_Code.sas)** |
| All attributes | Inconsistent delimiter | Due to the presence of unwanted semi-colons, the document was not easily readable. We fixed that by splitting the string and inserting the data into the respective columns as required **(PECproduct\_Code.sas)** |
| unitCost | Missing unitCost | There was column shifting. To fix this, we shifted them by finding the presence of a dot in the string and splitting the string based on that **(PECproduct\_Code.sas)** |
| supplierName | Missing supplierName | The supplierName PEC was missing and we replaced it using the missing method in SAS **(PECproduct\_Code.sas)** |
| prodDescription | Inconsistent values | Used the tranwrd method in SAS to replace these occurrences – (Equip, Equipment) **(PECproduct\_Code.sas)** |
| unitCost | Missing values | Calculated the unit cost using manufacturing\_cost from PEC\_Manufacturing and qty from PEC\_Invoice. (PECproduct\_Transformation.ktr) |
| **TPCWcustomer.csv** | All attributes | Extra spaces and quotes | Removed extra spaces and quotes using SAS code **(TPCWcustomer\_Code.sas)** |
| All attributes | Inconsistent delimiter | Due to the presence of unwanted semi-colons, the document was not easily readable. We fixed that by splitting the string and inserting the data into the respective columns as required **(TPCWcustomer\_Code.sas)** |
| custtype, address, name | Inconsistent values | Used the tranwrd method in SAS to replace these occurrences – (STATELOCALGOVT, COMERCIAL, St., Ave., Inc., Co., etc.) **(TPCWcustomer\_Code.sas)** |
| Zip | Zip format missing leading 0 if it is of length 4 | Fixed this using the concatenation operation using || after using the strip and scan function to slice the string and concatenate it **(TPCWcustomer\_Code.sas)** |
| Address | Addr1 and Addr2 split | Split the single attribute into two by using the scan method of SAS using a delimiter **(TPCWcustomer\_Code.sas)** |
| Address | Addr1 and Addr2 split | Split the single attribute into two without using a delimiter – We checked if the first character was a numerical value. If not, we extracted the first two words from the address and inserted them into addr2 **(TPCWcustomer\_Code.sas)** |
| customer\_type | Convert to upper case | Convert the customer\_type to upper case  **(TPCCustomer\_Transformation.ktr)** |
| **TPCWinvoice.csv** | OrderDate, SalesDate | Format inconsistencies | The date field was in the form of both MM-DD-YY or MM/DD/YY. We fixed this by splitting the string and then rejoining them to get the data in a uniform format **(TPCWinvoice\_Code.sas)** |
| custID | Negative CustID | There were negative customer IDs which we fixed by checking the first character of the string, which, if found to be “-“, use the absolute value of the string **(TPCWinvoice\_Code.sas)** |
| Discounted | Empty discounted row | There was a row with missing discounted value. This was fixed by sorting by invoiceID and replacing the missing value with the duplicate invoiceID’s discounted attribute value **(TPCWinvoice\_Code.sas)** |
|  | Amt | Calculate the correct amount | Used javascript to calculate the amount using the price1,price2, qty, and discount (TPCWInvoice\_Transformation.ktr) |
| **TPCWproduct.csv** | All attributes | Extra spaces and quotes | Removed extra spaces and quotes using SAS code **(TPCWproduct\_Code.sas)** |
| All attributes | Inconsistent delimiter | Due to the presence of unwanted semi-colons, the document was not easily readable. We fixed that by splitting the string and inserting the data into the respective columns as required **(TPCWproduct\_Code.sas)** |
| supplierAttn | Remove the prefix “Attn: “ | Used the tranwrd method in SAS to replace these occurrences with empty strings **(TPCWproduct\_Code.sas)** |
| productName, supplierName, supplierAddress | Inconsistent Abbreviations | Used the tranwrd method in SAS to replace these occurrences – (Equip, Inc., Ave., St.) **(TPCWproduct\_Code.sas)** |
| supplier\_State | Convert to upper case | Convert the supplier\_State to upper case  **(TPCProduct\_Transformation.ktr)** |
| All attributes | Remove duplicates | To remove duplicates using nodupkey in SAS **(TPCWproduct\_Code.sas)** |

**Data Transformation**

Data Transformation was used to transform the data to desired dimension tables for the warehouse. We merged the PEC and TPC files using steps like Merge step, Join step, etc. We also added various attributes for distinguishing between the TPCW and PEC data. Apart from that we also created temporary tables for Fact Table before loading them into the database. The below table presents the transformation for each dimension along with the problem and their respective code files.

***Data Transformation***

|  |  |
| --- | --- |
| **DM Table** | **Image Creation Process (attach code)** |
| **Customer Dimension** | Merged the Customer Types from PEC and TPC into 1 CSV file.  Performed a join operation on customer type with PEC Customer and a join operation on customer type with TPC Customer and then merged the two joined tables into 1 CSV. We also included a new column to store information about the source of the data called customer\_sourceflag. **(Merged\_Customer\_Type\_Transformation.ktr,**  **Merged\_Customer\_Transformation.ktr)** |
| **Product Dimension** | Merged the Business Units from PEC and TPC into 1 CSV file.  Merged the Product Types from PEC and TPC into 1 CSV file.  Performed a join operation on PECproduct with the business unit, and product type and Performed a join operation on TPCWproduct with the business unit, and product type and merged the 2 joined tables into 1 CSV. We also included a new column to store information about the source of the data called product\_sourceflag.  **(Merged\_Business\_Unit\_Transformation.ktr, Merged\_Product\_Type\_transformation.ktr, Merged\_Product\_Transformation.ktr)** |
| **Supplier Dimension** | Extracted the supplier information from TPCWproduct and PECproduct. These were merged into one single CSV file using Pentaho.  **(Supplier\_Transformation.ktr)** |
| **Checkout Details Junk Dimension** | Took the distinct values from PECinvoice. We got all the cartesian products for Order Method, Shipping Method, and Payment Method. These values were then inserted into a CSV file.  **(Junk\_Method\_Transformation.ktr)** |
| **Order Date Dimension** | Extracted the Order dates from PECinvoice. These values were inserted into a CSV file.  **(Order\_Date\_Transformation.ktr)** |
| **Sales Date Dimension** | Extracted the Sales dates from TPCWinvoice and PECinvoice. These values were merged and then inserted into a CSV file.  **(Sales\_Date\_Transformation.ktr)** |
| **Financial Control Analysis Fact** | Created temporary files to store information regarding TPCWinvoice and PECinvoice. These files were a result of joining the attributes mentioned before with their respective Product information. Performed calculations to get the facts like cost\_price, gross\_profit, and shipping\_time. Another column called sale\_type was included to denote if the transaction was performed using PEC or TPCW.  **(TPCWinvoice\_Temporary\_Transformation.ktr, PECinvoice\_Temporary\_Transformation.ktr)** |

**Data Loading**

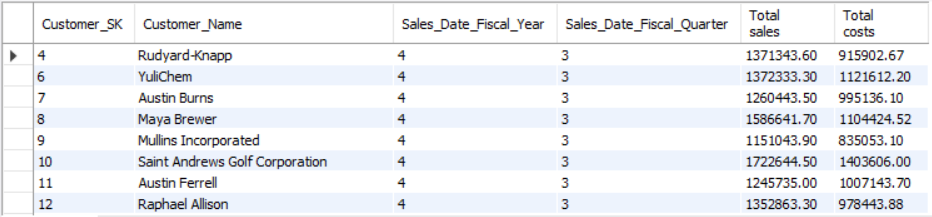
For the final step, we loaded the data into the dimensions and fact table using transformations and SQL queries. We used SQL queries for adding the NULL record in the Dimensions. We also used temporary tables for the loading data into the Financial\_Control\_Analysis\_Fact. The two SQL queries read the data from temporary tables and get the SK’s from the dimension tables. Once all the data was loaded we created the foreign key relations between the Fact and Dimension.

|  |  |
| --- | --- |
| **DM Table** | **Table Population Process (attach code)** |
| Customer\_Dim | Pentaho was used to load the data with the help of a Table output step  Added null record to the dimension.  **(Merged\_Customer\_Transformation.ktr, Insert\_null\_records.sql)** |
| Product\_Dim | Pentaho was used to load the data with the help of a Table output step.  Added null record to the dimension  **(Merged\_Product\_Transformation.ktr, Insert\_null\_records.sql)** |
| Supplier\_Dim | Pentaho was used to load the data with the help of a Table output step.  Added null record to the dimension  **(Merged\_Supplier\_Transformation.ktr, Insert\_null\_records.sql)** |
| Checkout\_Details\_Junk\_Dim | Pentaho was used to load the data with the help of a Table output step  Added null record to the dimension.  **(Junk\_Method\_Transformation.ktr, Insert\_null\_records.sql)** |
| Order\_Date\_Dim | Pentaho was used to load the data with the help of a Table output step.  Added null record to the dimension.  **(Order\_Date\_Transformation.ktr, Insert\_null\_records.sql)** |
| Sales\_Date\_Dim | Pentaho was used to load the data with the help of a Table output step.  Added null record to the dimension.  **(Sale\_Date\_Transformation.ktr, Insert\_null\_records.sql)** |
| Financial\_Control\_Analysis\_Fact | Used temporary tables mentioned above to select the attributes required for the fact table. SQL queries were used to define the foreign keys and then, they were used to load the data to the fact table where we had extensive joins between the dimensional tables and the temporary tables to deliver the desired attributes of the fact table.  **(TPCWinvoice\_Temporary\_Transformation.ktr, PECinvoice\_Temporary\_Transformation.ktr, Insert\_fact\_records.sql, Create\_Foreign \_Keys.sql)** |

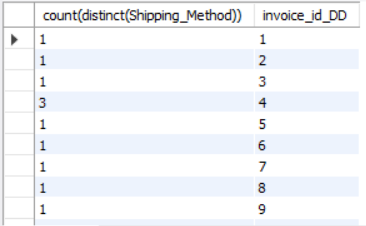
**SQL Queries**

We chose three queries for analyzing the data warehouse. For the reporting queries we used indexes for the optimized query results.

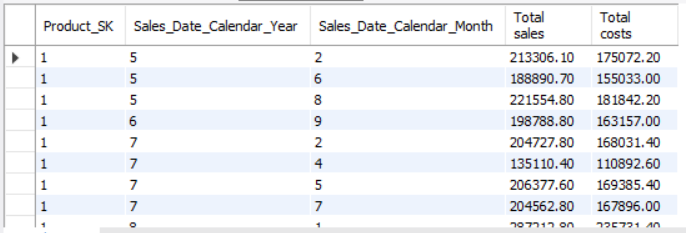
1. A report that shows the sales, and costs associated with each customer every quarter.



1. The maximum number of different shipping methods on each invoice?



1. A report that shows the sales, and costs associated with each product every month.



**Assumptions**

During the ETL process we made the following assumptions:

1. In TPCinvoice there is one row with just invoiceID followed by headers, which we have deleted using SAS.
2. In TPCinvoice there is a row with missing discounted value. We went through the file using SAS to check the value for the discounted attribute for that particular invoiceID. We assigned the same value to the row with the missing discounted value.
3. We saw that in PECProduct.csv, there is one record that has a Product Type ID of 33 assigned to it. We looked at the files we have and we found that there is no Product Type with ID 33. We hard-coded this value after consultation with Prof. [Jai Kang](mailto:jwkics@rit.edu).
4. We found that there are some prodID (101,102,399) in TPCWinvoice which are not present in TPCProduct(ProdID 101, 102 were removed during deduplication in SAS). There are about 1700 rows in the TPCWinvoice that belong to this prodID. To fix this, we deleted those IDs and the entries for the respective products in the transformed file.
5. When we merged TPCWproduct and PECproduct, there were multiple entries with the same prod\_id, but different information in the other attributes. We weren't sure how to handle them. We then made use of a flag that denoted the source of the data (PEC or TPCW). This way, it becomes easier for the users to fetch the value according to their requirements.
6. Because TPCW does not have an order date and thus, shipping time and cost are to be calculated, we have assumed that they are NULL.
7. After filtering and cleaning the TPCWinvoice file to create the Supplier dimension, we found that some rows had almost similar values in all columns except for a missing/added letter in one of the attributes. We have not removed them as we do not have a lookup to refer and correct these values.

**Conclusion**

In this Lab, we understood that cleaning the data before transforming and loading it is an important step. It was one of the most time-consuming steps. After completing each cleaning step we were able to find new issues and had to resolve them. At the end of each step, we had a much better understanding of the issue and how to fix them. We also reached out to the professor a couple of times with the issues. As suggested by the professor, we started early to complete the lab on time. The assignment was interesting as we got a chance to explore the various functions of SAS and Pentaho PDI.