Business Intelligence

# STS Sales Report

Daniel Pustan

31st December, 2021

Halmstad University

# Table of contents

# 1. Introduction

In this laboratory report, I will present the answer to five questions which aim to support strategic decisions to improve the sales performance for STS company. These answers are the result of an elaborated process of Extraction, Transformation and Loading (ETL) of data into a data mart and the subsequent analysis using the Qliksense tool.

The database analyzed belongs to STS, an organization that sells three lines of articles (telecom, electronics and sports) in 30 cities around the world. The final work takes the form of a dashboard with filter panes providing a clear overview of the most relevant indicators in the database as well as the possibility to drill-down for more targeted insights.

The theory and methodology draws on the book *Business Intelligence and Analytics*: Systems for Decision Support (Turban et al, 2014).

The remaining of the report is structured in four sections. First, the theory and methodology are presented. Then the results are shown followed by a discussion and finally the conclusions are laid out.

# 2. Theory and methodology

In this chapter, I will start by defining some key concepts relevant to this process such as ETL, data warehouse, data mart and dashboards and then I will discuss the implementation more into detail.

## Extraction, Transformation and Load (ETL)

The ETL process represents the core of any data warehousing project and it is probably the most time consuming depending on the quality of the data. It consists of *extraction* which refers to reading data from all relevant sources, *transformation where* the extracted data is converted into the desired form) and *load* where the data is moved into the data warehouse. (Turban et al, 2014).

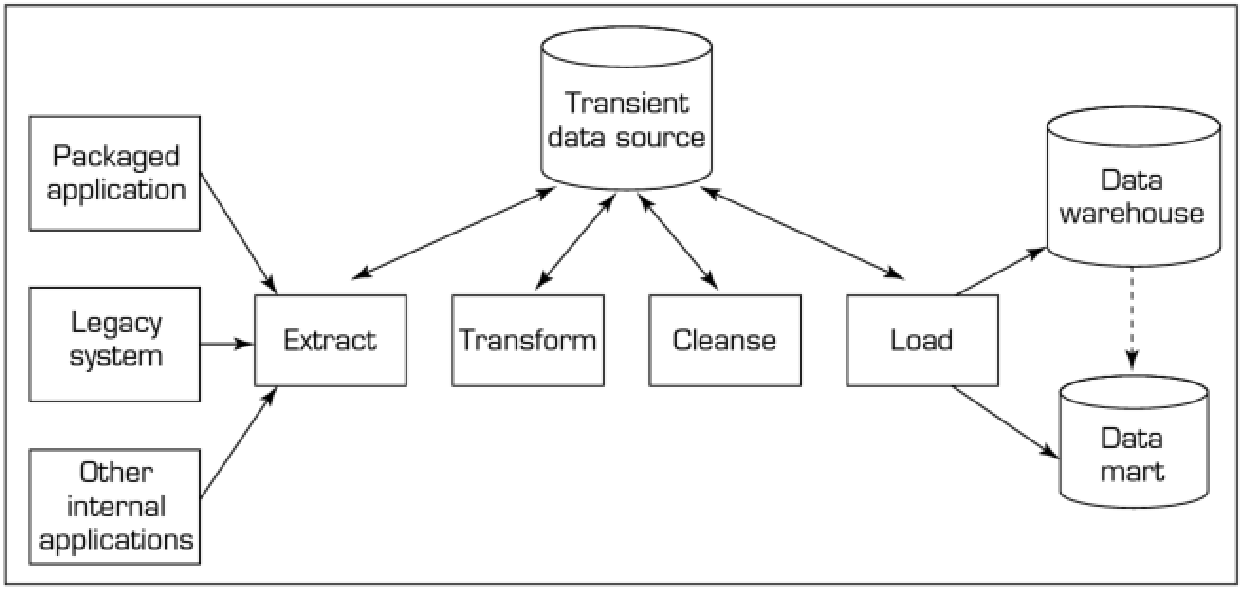


**Figure 1** The ETL process (Source: Turban et al, 2014, p. 130)

## Data Warehouse and Data Mart

For the development of the data warehouse, I use the Kimball model also known as the data mart approach. A data mart is a reduced version of a data warehouse focused on the needs of a specific department or area of interest. (Turban et al, 2014)

The most common implementation of dimensional modeling for a data warehouse is the star schema which was also used in this project. According to Adamson (2009) as cited by Turban et al (2014, p.138) a star schema “contains a central fact table surrounded by and connected to several dimension tables”. The fact table contains the descriptive attributes such as operational metrics, performance and aggregated measures as well as the foreign keys which link with the dimension tables. The dimension tables comprise of attributes describing the data in the fact table. (Turban et al, 2014)

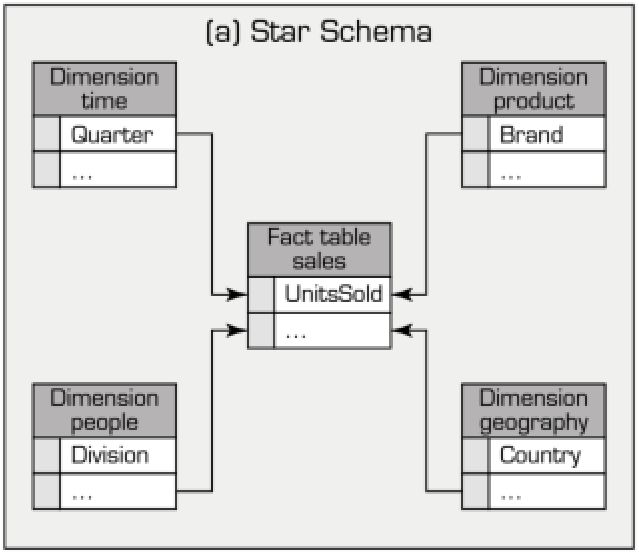


**Figure 2** Star schema (Source: Turban et al, 2014, p.139)

## Performance Dashboard

A visual display of the most relevant information into a single screen so that it may be easy to read and drilled in for further exploration (Turban et al, 2014). According to Eckerson (2011), it comprises three layers of information: *monitoring* (graphical abstract data to monitor KPI), *analysis* (brief dimensional data to analyze the root cause) and *management* (detailed operational data identifying the actions necessary to resolve an issue).

## Implementation

First, the database is analyzed to get an idea regarding the most useful information which can be derived and build the data mart accordingly. Hence, the following five questions will be addressed in this project:

**1. How did profit vary over time?**

This is a key metric that shows the overall performance of the company.

**2. Which are the most profitable cities?**

The answer to this question can give an idea of successful strategies to be extended to the less profitable cities.

**3. Who are the most profitable customers?**

The answer to this question can support the marketing strategies of the company.

**4. Which are the most profitable products?**

This question offers an insight on the areas to invest in and the direction for development..

**5. In what periods do most shipment delays occur?**

Answering this question may help the company to adapt their HR capacities balancing customer satisfaction through on-time delivery and the salary costs by using extra personnel when the situation requires.

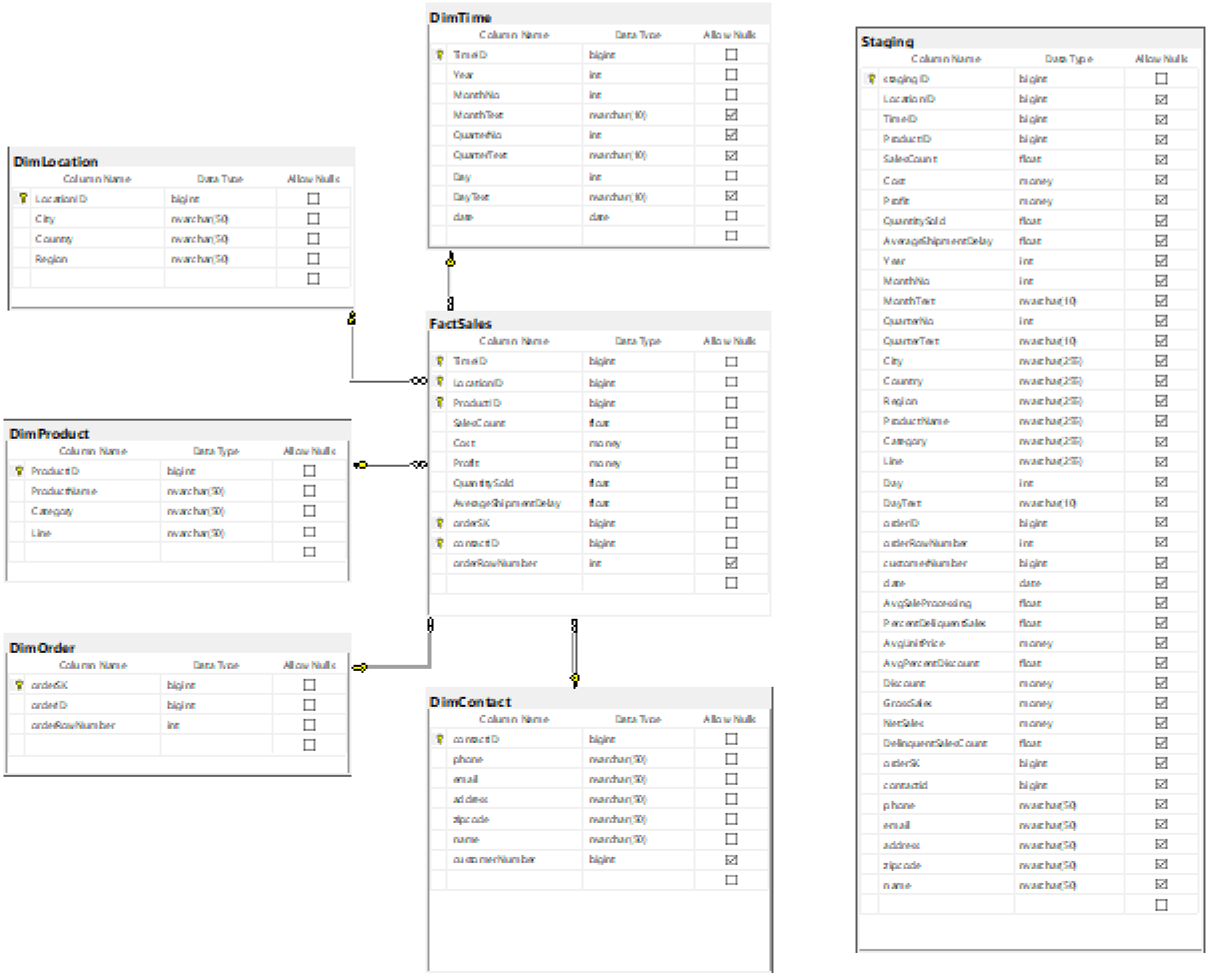
The questions purposely target the highest level of granularity (e.g. city or product) in each dimension to provide increased flexibility when interacting with the dashboard and consequently allow for a  broader spectrum of insights.

Once the questions were articulated, I created a data map to facilitate keeping track of source and destination data.

| Source | | Destination | |
|---|---|---|---|
| **Column** | **Data type** | **Column** | **Data type** |
| STSOrderRow.[SalesCount] | float | Staging.[SalesCount] | float |
| STSOrderRow.[Cost] | money | Staging.[Cost] | money |
| STSOrderRow.[Profit] | money | Staging.[Profit] | money |
| STSOrderRow.[QuantitySold] | float | Staging.[QuantitySold] | float |
| STSOrderRow.[AverageShipmentDelay] | float | Staging.[AverageShipmentDelay] | float |
| STSCity.[City] | nvarchar(255) | Staging.[City] | nvarchar(50) |
| STSCity.[Country] | nvarchar(255) | Staging.[Country] | nvarchar(50) |
| STSCountry.[Region] | nvarchar(255) | Staging.[Region] | nvarchar(50) |
| STSProduct.[ProductName] | nvarchar(255) | Staging.[ProductName] | nvarchar(50) |
| STSCategory.[Category] | nvarchar(255) | Staging.[Category] | nvarchar(50) |
| STSLine.[Line] | nvarchar(255) | Staging.[Line] | nvarchar(50) |
| STSOrderRow.[orderRowNumber] | int | Staging.[orderRowNumber] | int |
| STSOrderRow.[orderID] | bigint | Staging.[orderID] | bigint |
| STSOrder.[date] | date | Staging.[date] | date |
| STSOrderRow.[AvgSaleProcessing] | float | Staging.[AvgSaleProcessing] | float |
| STSOrderRow.[PercentDeliquentSales] | float | Staging.[PercentDeliquentSales] | float |
| STSOrderRow.[AvgUnitPrice] | money | Staging.[AvgUnitPrice] | money |
| STSOrderRow.[AvgPercentDiscount] | float | Staging.[AvgPercentDiscount] | float |
| STSOrderRow.[Discount] | money | Staging.[Discount] | money |
| STSContact.[phone] | nvarchar(50) | Staging.[phone] | nvarchar(50) |
| STSContact.[email] | nvarchar(50) | Staging.[email] | nvarchar(50) |
| STSContact.[address] | nvarchar(50) | Staging.[address] | nvarchar(50) |
| STSContact.[zipcode] | nvarchar(50) | Staging.[zipcode] | nvarchar(50) |
| STSContact.[name] | nvarchar(50) | Staging.[name] | nvarchar(50) |

**Table 1** Data map

As mentioned before, the data mart was built based on a star schema comprising one FactSales table and five dimension tables: Time, Location, Product, Order and Contact. I connected the fact table with the five dimension tables by assigning and connecting a primary key in each dimension with the five surrogate keys in the fact table. In addition, I created a Staging table to act as a bridge between STSSales database and the final data mart.



**DimTime**

| | Column Name | Data Type | Allow Nulls |
|---|---|---|---|
| 🔑 | TimeID | bigint | □ |
| | Year | int | □ |
| | MonthNo | int | □ |
| | MonthText | nvarchar(10) | ☑ |
| | QuarterNo | int | ☑ |
| | QuarterText | nvarchar(10) | ☑ |
| | Day | int | □ |
| | DayText | nvarchar(10) | ☑ |
| | date | date | □ |
| | | | □ |

**DimLocation**

| | Column Name | Data Type | Allow Nulls |
|---|---|---|---|
| 🔑 | LocationID | bigint | □ |
| | City | nvarchar(50) | □ |
| | Country | nvarchar(50) | □ |
| | Region | nvarchar(50) | □ |
| | | | □ |

**FactSales**

| | Column Name | Data Type | Allow Nulls |
|---|---|---|---|
| 🔑 | TimeID | bigint | □ |
| 🔑 | LocationID | bigint | □ |
| 🔑 | ProductID | bigint | □ |
| | SalesCount | float | □ |
| | Cost | money | □ |
| | Profit | money | □ |
| | QuantitySold | float | □ |
| | AverageShipmentDelay | float | □ |
| 🔑 | orderSK | bigint | □ |
| 🔑 | contactID | bigint | □ |
| | orderRowNumber | int | ☑ |
| | | | □ |

**DimProduct**

| | Column Name | Data Type | Allow Nulls |
|---|---|---|---|
| 🔑 | ProductID | bigint | □ |
| | ProductName | nvarchar(50) | □ |
| | Category | nvarchar(50) | □ |
| | Line | nvarchar(50) | □ |
| | | | □ |

**DimOrder**

| | Column Name | Data Type | Allow Nulls |
|---|---|---|---|
| 🔑 | orderSK | bigint | □ |
| | orderID | bigint | □ |
| | orderRowNumber | int | □ |
| | | | □ |

**DimContact**

| | Column Name | Data Type | Allow Nulls |
|---|---|---|---|
| 🔑 | contactID | bigint | □ |
| | phone | nvarchar(50) | □ |
| | email | nvarchar(50) | □ |
| | address | nvarchar(50) | □ |
| | zipcode | nvarchar(50) | □ |
| | name | nvarchar(50) | □ |
| | customerNumber | bigint | ☑ |
| | | | □ |

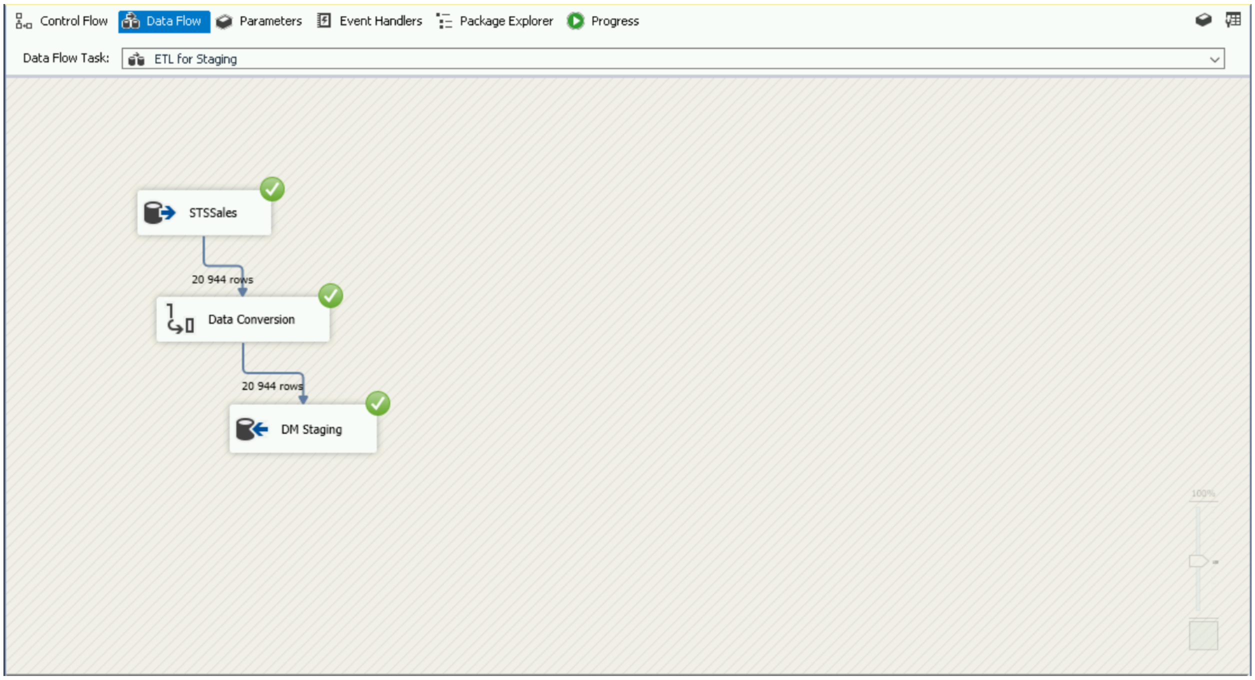
**Staging**

| | Column Name | Data Type | Allow Nulls |
|---|---|---|---|
| 🔑 | stagingID | bigint | □ |
| | LocationID | bigint | ☑ |
| | TimeID | bigint | ☑ |
| | ProductID | bigint | ☑ |
| | SalesCount | float | ☑ |
| | Cost | money | ☑ |
| | Profit | money | ☑ |
| | QuantitySold | float | ☑ |
| | AverageShipmentDelay | float | ☑ |
| | Year | int | ☑ |
| | MonthNo | int | ☑ |
| | MonthText | nvarchar(10) | ☑ |
| | QuarterNo | int | ☑ |
| | QuarterText | nvarchar(10) | ☑ |
| | City | nvarchar(255) | ☑ |
| | Country | nvarchar(255) | ☑ |
| | Region | nvarchar(255) | ☑ |
| | ProductName | nvarchar(255) | ☑ |
| | Category | nvarchar(255) | ☑ |
| | Line | nvarchar(255) | ☑ |
| | Day | int | ☑ |
| | DayText | nvarchar(10) | ☑ |
| | orderID | bigint | ☑ |
| | orderRowNumber | int | ☑ |
| | customerNumber | bigint | ☑ |
| | date | date | ☑ |
| | AvgSaleProcessing | float | ☑ |
| | PercentDelinquentSales | float | ☑ |
| | AvgUnitPrice | money | ☑ |
| | AvgPercentDiscount | float | ☑ |
| | Discount | money | ☑ |
| | GrossSales | money | ☑ |
| | NetSales | money | ☑ |
| | DelinquentSalesCount | float | ☑ |
| | orderSK | bigint | ☑ |
| | contactid | bigint | ☑ |
| | phone | nvarchar(50) | ☑ |
| | email | nvarchar(50) | ☑ |
| | address | nvarchar(50) | ☑ |
| | zipcode | nvarchar(50) | ☑ |
| | name | nvarchar(50) | ☑ |
| | | | □ |

**Figure 3** Data mart diagram

In the following steps, I ran reiterated ETL processes for each table to populate the data mart with data. Each ETL process is described in detail below.

**ETL for Staging**. In the Extraction step, all data is sourced from STSSales database. In the Transformation step, the length of QuarterName is reduced from 10 to 2 characters. Finally, the data is Loaded in the Staging table and columns are carefully mapped (for details please check Annex C).



Data Flow Task: ETL for Staging

**Figure 4** ETL for Staging

In the following flows, the Extraction is made from the Staging table using a different SQL command for each dimension which are detailed in Annex A of this report. Hence, in order to avoid redundancy the Extraction step will not be mentioned but only the Transformation and Loading.

**ETL for DimTime**. In the Transformation step, Data Conversion, I derived the Year, Month and Day using the expressions YEAR(date), MONTH(date) and DAY(date). In Data Conversion, I changed the length for the QuarterName variable from 10 to 2 characters. In the final step, I connected the flow to DimTime table and mapped the columns (for details please check Annex C).

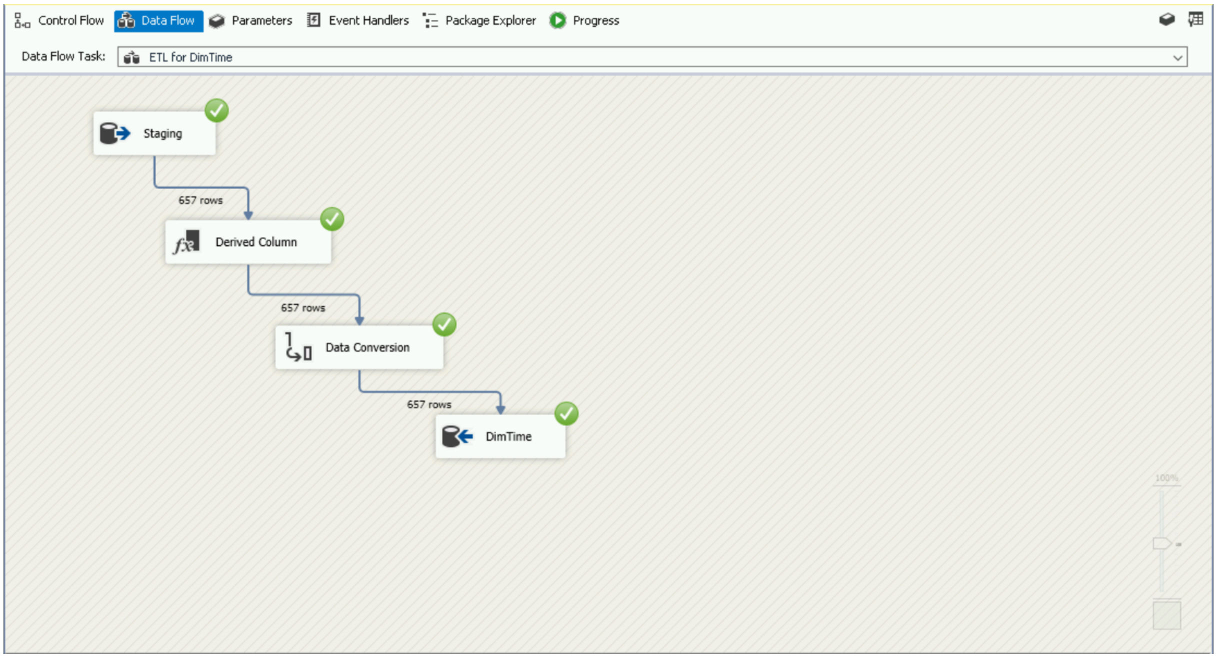


**Figure 5** ETL for DimTime

**ETL for DimLocation.** In the transformation step, the length of City, Country and Region are reduced to 50. Then the data is loaded in DimLocation table after mapping the columns (for details please check Annex C).

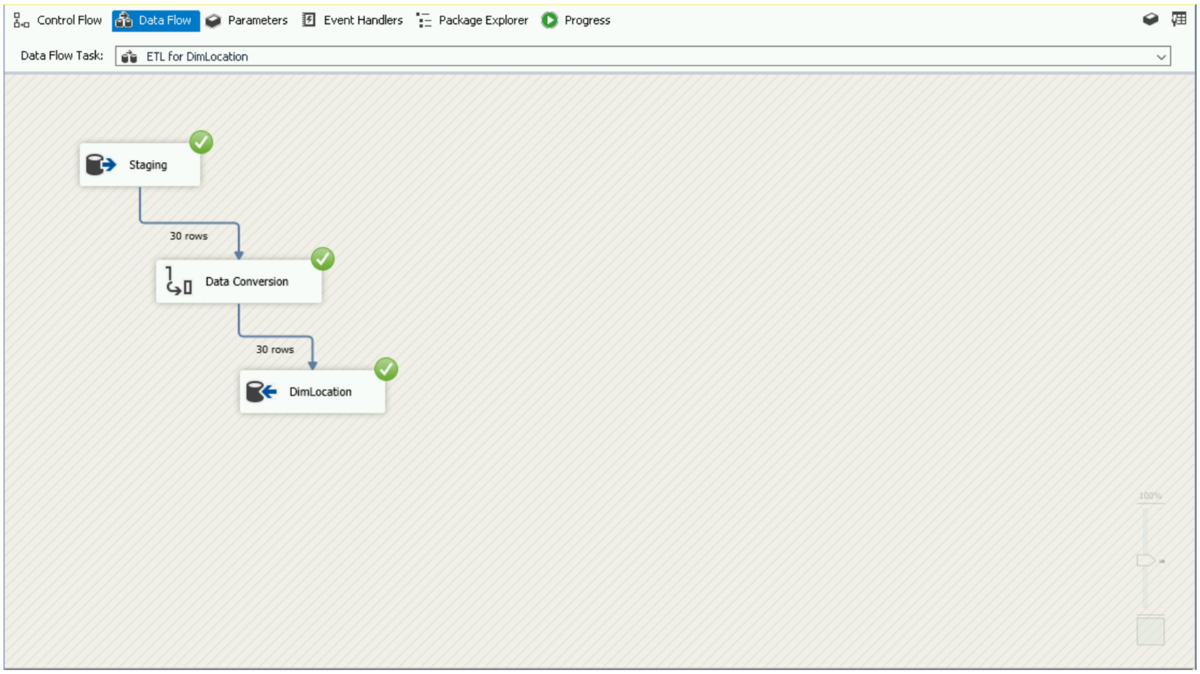
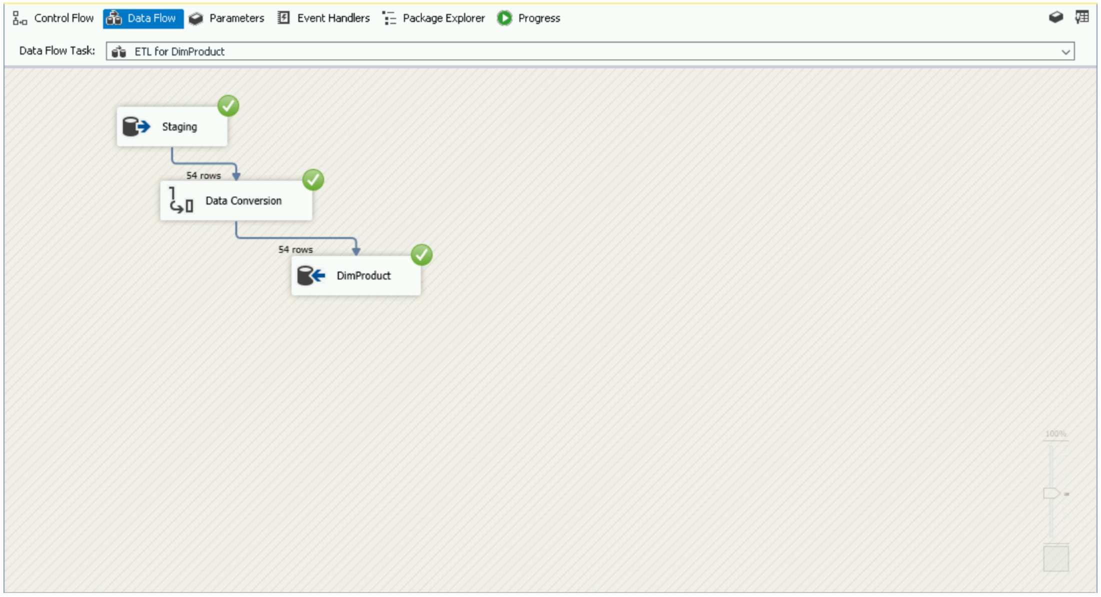


**Figure 6** ETL for DimLocation

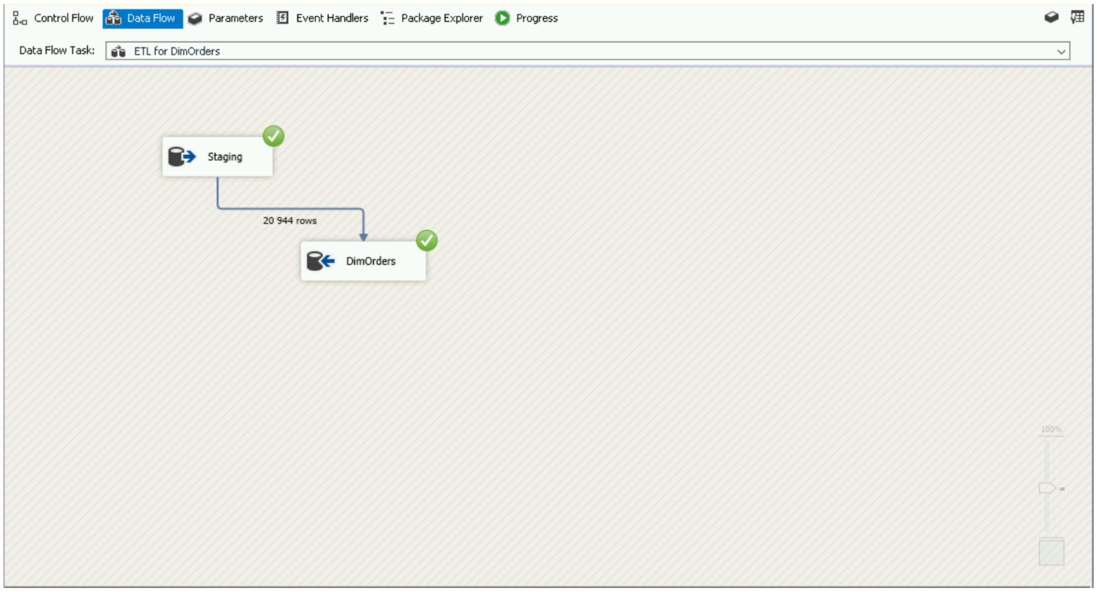
**ETL for DimProduct.** Here the length of ProductName, Category and Line is Transformed to 50. Then the data is loaded into DimProduct table and the columns are mapped (for details please check Annex C).



Data Flow Task: ETL for DimProduct

**Figure 7** ETL for DimProduct

**ETL for DimOrders.** No transformation was performed. The data was loaded in the DimOrder table after mapping the columns (for details please check Annex C).



Data Flow Task: ETL for DimOrders

**Figure 8** ETL for DimOrders

**ETL for DimContact.** Just like the previous process, no transformation is performed on this data. The data is loaded in DimContact table and mapping is performed on columns (for details please check Annex C).

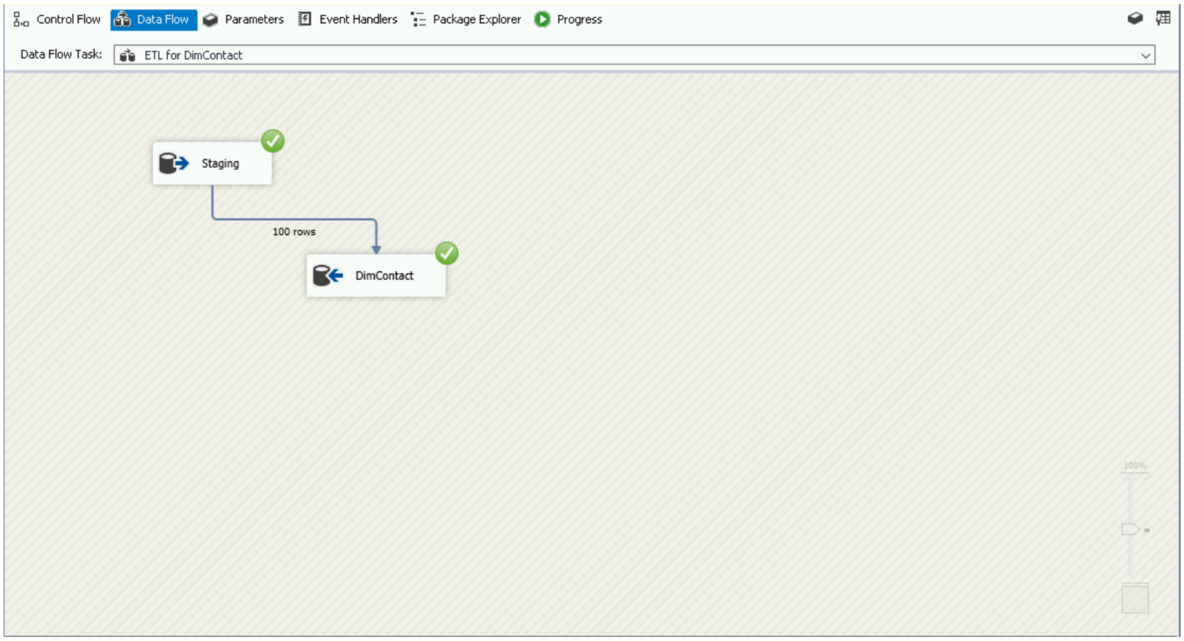


**Figure 9** ETL for DimContact

**ETL for FactSales.** No SQL command is used for the extraction of data. It is accessed by connecting directly to the Staging table. Then a series of Lookup actions are performed to identify the correct keys once the columns displaying the highest granularity are connected. Finally, the mapping is performed on the columns and the data is loaded (for details please check Annex C).

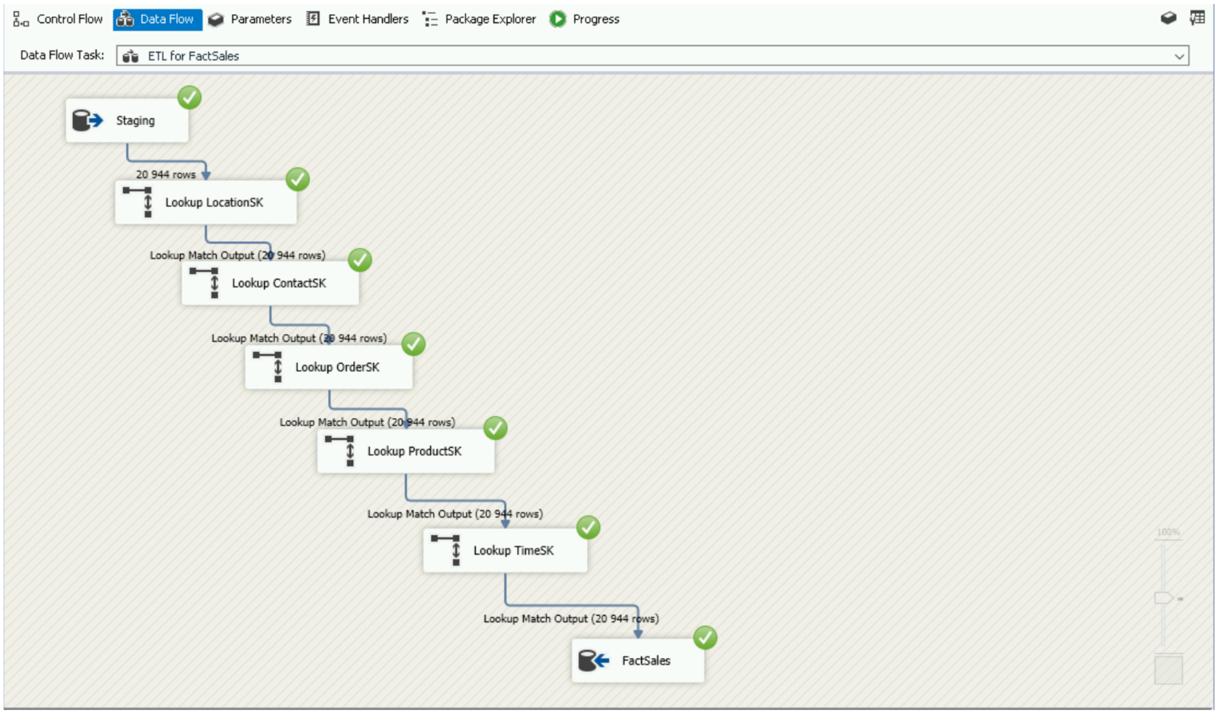


**Figure 10** ETL for FactSales

Once all steps of the process were completed, I ran the entire ETL process and also timed its duration which was under six seconds.

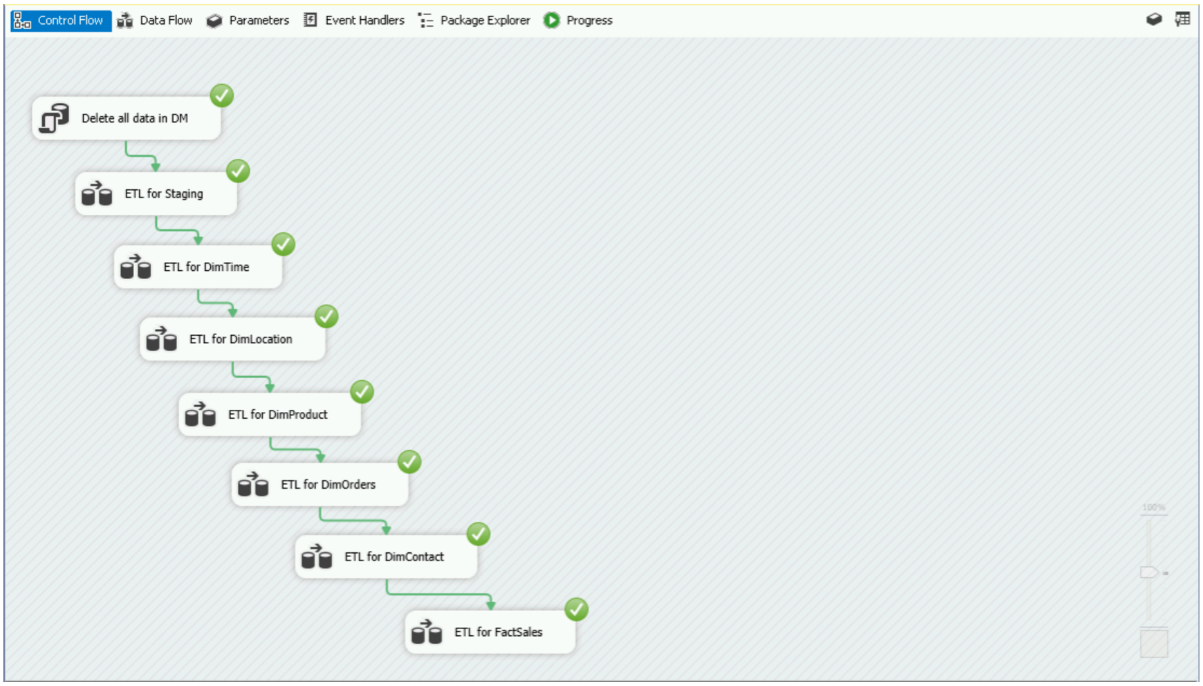


**Figure 11** ETL process

After completing the ETL process, I am ready to use the data mart created to provide an answer to the questions raised in the form of a dashboard. I do that by loading the data mart in Qlik Sense visualisation tool. Then I built the dashboard that would answer the five questions and would become the first BI tool for STS company.

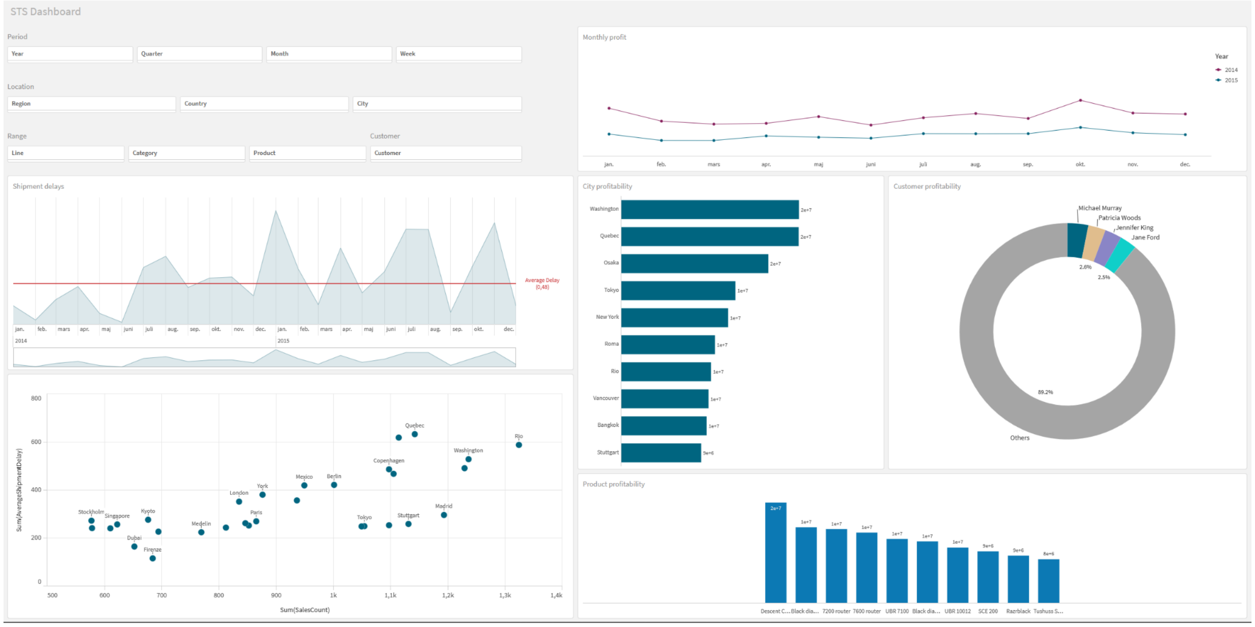


**Figure 12** STS Dashboard

# 3. Result

Figure 13 answers the first question: "*how did profit vary over time?*". From the graph we see that profit was significantly lower in 2015 than it was in 2014. This is a first indicator that deeper analysis is required to identify what could have caused the variation in profit between the two years.

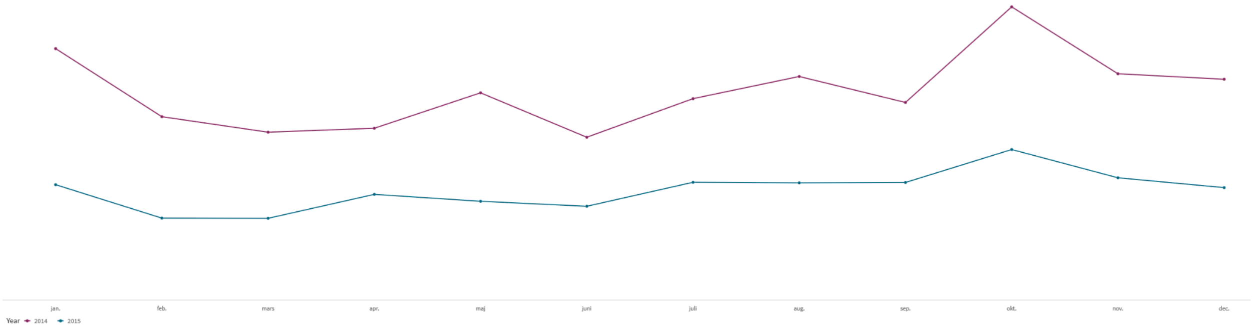


**Figure 13** Monthly profit

Maybe the fastest way to get on the path of success could be by replicating the already existing success stories from other cities. This is the reason why the second question is raised: "which are the most profitable cities"? The figure reveals that cities in the USA and Japan are the most profitable while no city from the EEMEA region is present in the top 10.

It is worth mentioning though that there are important variation factors between cities (e.g. weather, customer behavior, etc.) which need to be taken into account when considering applying the recipe of success to another city.

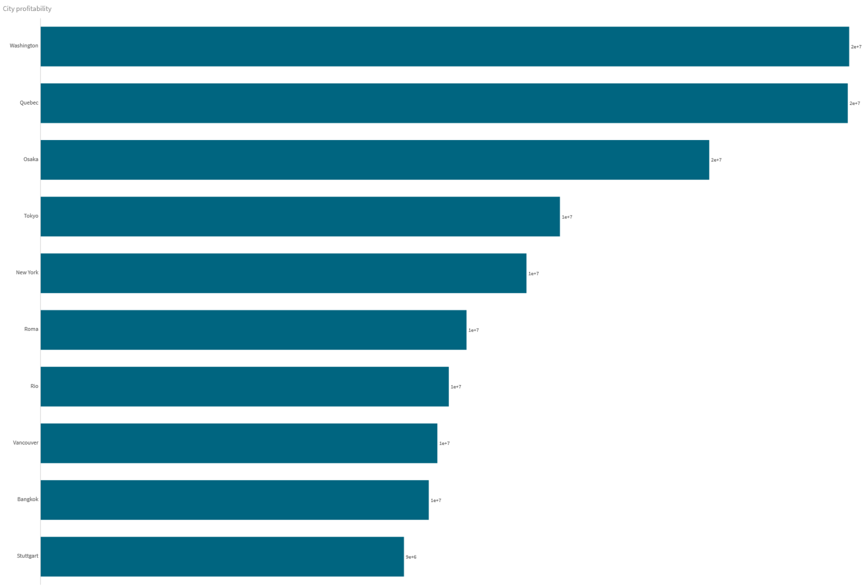


**Figure 14** City profitability

Another strategy with potential to increase sales is focusing on customers. Identifying the most profitable customers, their contribution to the total profitability as well as the products these customers buy may be a good start for market segmentation and targeted campaigns. Hence, figure 15 answers the third question: "who are the most profitable customers?". Thus, we discover that the top 5 customers bring 13% of the company's profit with shares between 3.1% and 2.3%.

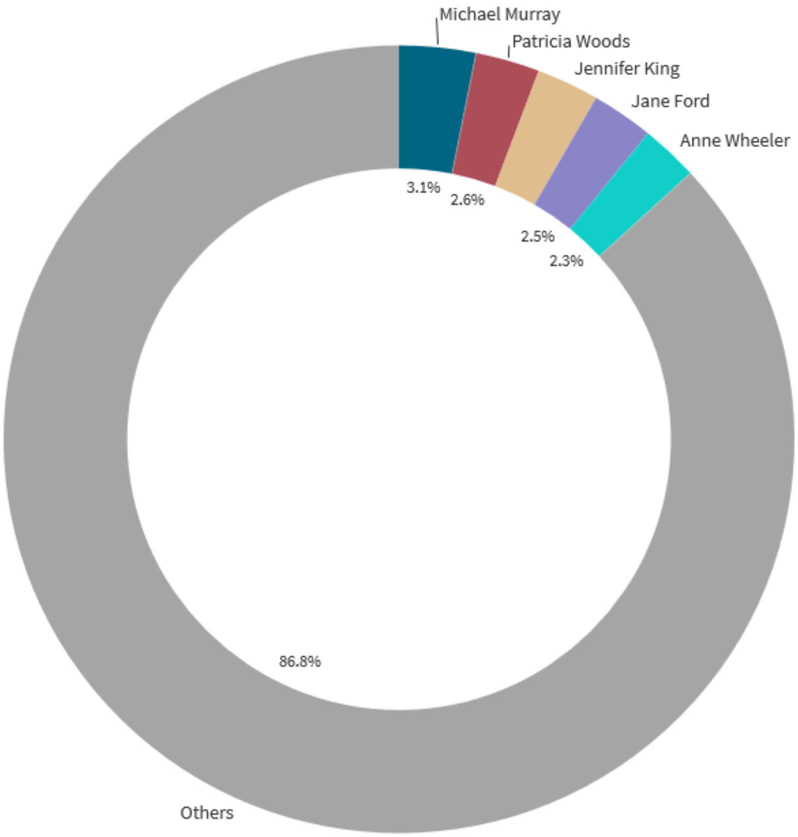


**Figure 15** Customer profitability

The fourth question: *“which are the most profitable products?”* is intended to support the company's investment decisions as well as provide insights about customer preferences when explored for different regions through filter panes. In figure 16 we can see that 8 products in the top 10 belong to telecom line and 2 to sports.

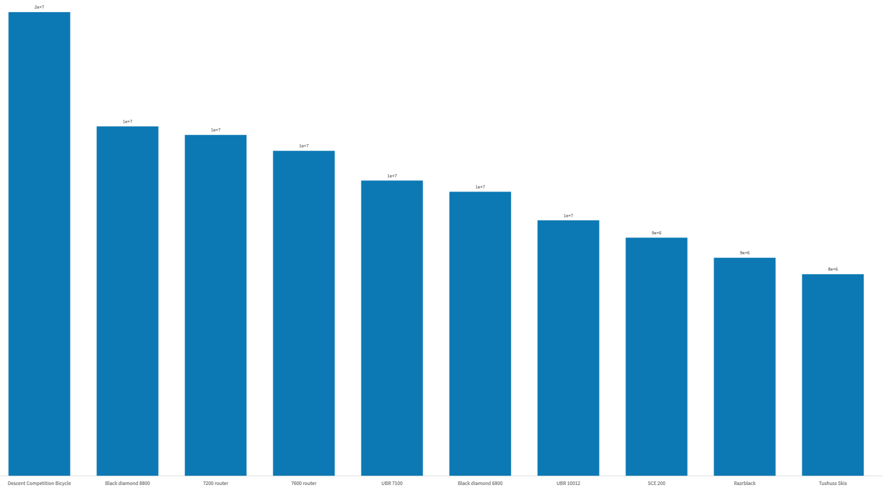


**Figure 16** Product profitability

The last question: *“in what periods do most shipping delays occur?”* is answered in figure 17 and is intended to provide an overview regarding the workload and potential bottlenecks which may affect customer satisfaction and profit if not solved promptly. It is worth noting that 2015 when the profit was lower displays higher peaks than the previous year, as well. This may be an indicator that shipping delays might affect profit to some extent.

The four peaks may be explained by Christmas Season (December - January), Easter (April), Summer holidays when usually there is less staff on prems (June - August) and Black Friday (November). It might be that the company does not cover very well these periods with high fluctuation of personnel.

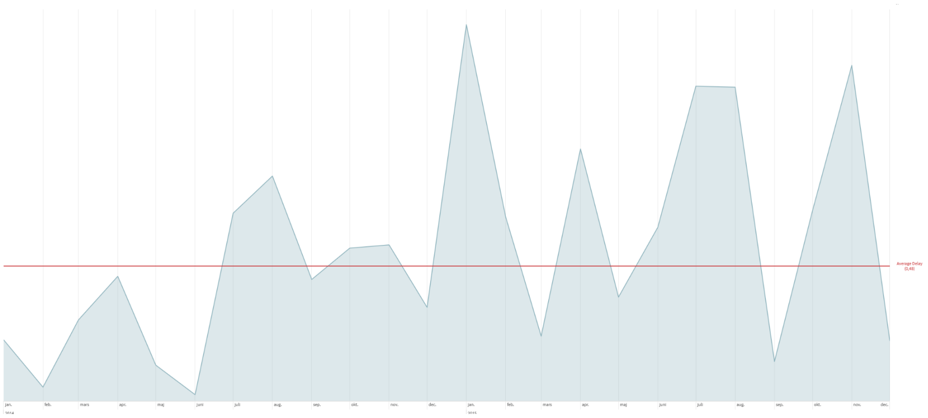


**Figure 17** Shipment delays

# 4. Conclusion

This assignment allowed me to go through all stages of a BI project from the definition of objectives to the design of the dashboard.  The experience reinforced the idea that the quality of data is of utmost importance in the process as the report insights are as good as the data on which they rely (“garbage in, garbage out”).

In addition, I learned about the importance of defining a clear scope from the very beginning as well as identifying the organization’s present and future requirements. This can avoid a lot of redundant work along the way.

# 5. References

1. Turban, E., Sharda, R. and Delen, D., 2014. Business Intelligence and Analytics. 10th ed. Harlow, United Kingdom: Pearson Education Canada.

2. Eckerson, W., 2011. *Performance dashboards*. 2nd ed. Hoboken, N.J.: John Wiley & Sons.

# 6. Annex

## Annex A

### ETL for Staging: SQL code for data extraction

```
SELECT DISTINCT stsproduct.[product name],
          stsproduct.category,
          line,
          stscity.city,
          stscity.country,
          region,
          customernumber,
          [date],
          Datepart(yyyy, [date])            AS [year],
          'QuarterNo' = CASE
                    WHEN Datepart(mm, date) IN ( 1, 2, 3 ) THEN 1
                    WHEN Datepart(mm, date) IN ( 4, 5, 6 ) THEN 2
                    WHEN Datepart(mm, date) IN ( 7, 8, 9 ) THEN 3
                    WHEN Datepart(mm, date) IN ( 10, 11, 12 ) THEN 4
                   END,
          'QuarterName' = CASE
                     WHEN Datepart(mm, date) IN ( 1, 2, 3 ) THEN
                     'Q1'
                     WHEN Datepart(mm, date) IN ( 4, 5, 6 ) THEN
                     'Q2'
                     WHEN Datepart(mm, date) IN ( 7, 8, 9 ) THEN
                     'Q3'
                     WHEN Datepart(mm, date) IN ( 10, 11, 12 ) THEN
                     'Q4'
                    END,
          Datepart(mm, [date])              AS [MonthNo],
          Substring(Datename(mm, [date]), 1, 3) AS [MonthName],
          Datepart(dd, [date])             AS [DayNo],
          Substring(Datename(dw, [date]), 1, 3) AS [DayName],
          stsorder.orderid,
          [sales count],
          [avg sale processing time],
          [percent delinquent sales],
          [avg unit price],
          [quantity sold],
          [average shipment delay],
          [profit],
          [cost],
          [avg percent discount],
          [discount],
          [gross sales],
          [net sales],
          [delinquent sales count],
          [orderrownumber],
          contactid,
          firstname,
          lastname,
          phone,
          email,
          [address],
          zipcode,
          NAME
FROM   stsproduct,
    stscategory,
    stsorderrow,
    stsorder,
    stscity,
    stscountry,
    stscontact
WHERE  stsproduct.category = stscategory.category
    AND stsorderrow.[product name] = stsproduct.[product name]
    AND stsorderrow.orderid = stsorder.orderid
    AND stsorder.city = stscity.city
    AND stscity.country = stscountry.country
    AND [stscontact].[contactid] = stsorder.[customernumber];
```

### ETL for DimTime: SQL code for data extraction

```
SELECT DISTINCT 'QuarterNo' = CASE
                   WHEN Datepart(mm, date) IN ( 1, 2, 3 ) THEN 1
                   WHEN Datepart(mm, date) IN ( 4, 5, 6 ) THEN 2
                   WHEN Datepart(mm, date) IN ( 7, 8, 9 ) THEN 3
                   WHEN Datepart(mm, date) IN ( 10, 11, 12 ) THEN 4
                  END,
         'QuarterName' = CASE
                    WHEN Datepart(mm, date) IN ( 1, 2, 3 ) THEN
                    'Q1'
                    WHEN Datepart(mm, date) IN ( 4, 5, 6 ) THEN
                    'Q2'
                    WHEN Datepart(mm, date) IN ( 7, 8, 9 ) THEN
                    'Q3'
                    WHEN Datepart(mm, date) IN ( 10, 11, 12 ) THEN
```

```
                                  'Q4'
                                END,
                Substring(Datename(mm, [date]), 1, 3) AS [MonthName],
                Substring(Datename(dw, [date]), 1, 3) AS [DayName],
                date
FROM   staging;
```

### ETL for DimLocation: SQL code for data extraction

```
SELECT DISTINCT city,
                country,
                region
FROM   staging;
```

### ETL for DimProduct: SQL code for data extraction

```
SELECT DISTINCT productname,
                category,
                line
FROM   staging;
```

### ETL for DimOrders: SQL code for data extraction

```
SELECT DISTINCT orderid,
                orderrownumber
FROM   staging;
```

### ETL for DimContact: SQL code for data extraction

```
SELECT DISTINCT phone,
                email,
                address,
                zipcode,
              name,
                customernumber
FROM   staging;
```

## Annex B

**Table 2** ETL for Staging - mapping

| Input Column | Destination Column |
|---|---|
| <ignore> | stagingID |
| <ignore> | LocationID |
| <ignore> | TimeID |
| <ignore> | ProductID |
| Sales Count | SalesCount |
| Cost | Cost |
| Profit | Profit |
| Quantity Sold | QuantitySold |
| Average Shipment Delay | AverageShipmentDelay |
| year | Year |
| MonthNo | MonthNo |
| MonthName | MonthText |
| QuarterNo | QuarterNo |
| <ignore> | QuarterText |
| City | City |
| Country | Country |
| Product Name | ProductName |
| Category | Category |
| Line | Line |
| DayNo | Day |
| DayName | DayText |
| orderid | orderID |
| OrderRowNumber | orderRowNumber |
| customernumber | customerNumber |
| date | date |
| Avg Sale Processing Time | AvgSaleProcessing |
| Percent Delinquent Sales | PercentDeliquentSales |
| Avg Unit Price | AvgUnitPrice |
| Avg Percent Discount | AvgPercentDiscount |

| | |
|---|---|
| Discount | Discount |
| Gross Sales | GrossSales |
| Net Sales | NetSales |
| Delinquent Sales Count | DelinquentSalesCount |
| <ignore> | orderSK |
| contactid | contactid |
| phone | phone |
| email | email |
| address | address |
| zipcode | zipcode |
| name | name |
| Region | Region |

**Table 3** ETL for DimTime - mapping

| Input Column | Destination Column |
|---|---|
| <ignore> | TimeID |
| DerivedYear | Year |
| DerivedMonth | Month |
| MonthName | MonthText |
| QuarterNo | QuarterNo |
| <ignore> | QuarterText |
| DerivedDay | Day |
| DayName | DayText |
| date | date |

**Table 4** ETL for DimLocation - mapping

| Input Column | Destination Column |
|---|---|
| <ignore> | LocationID |
| Copy of City | City |
| Copy of Country | Country |
| Copy of Region | Region |

**Table 5** ETL for DimProduct - mapping

| Input Column | Destination Column |
|---|---|

| | |
|---|---|
| <ignore> | ProductID |
| Copy of ProductName | ProductName |
| Copy of Category | Category |
| Copy of Line | Line |

**Table 6** ETL for DimOrders - mapping

| Input Column | Destination Column |
|---|---|
| <ignore> | orderSK |
| orderid | orderID |
| orderrownumber | orderRowNumber |

**Table 7** ETL for DimContact - mapping

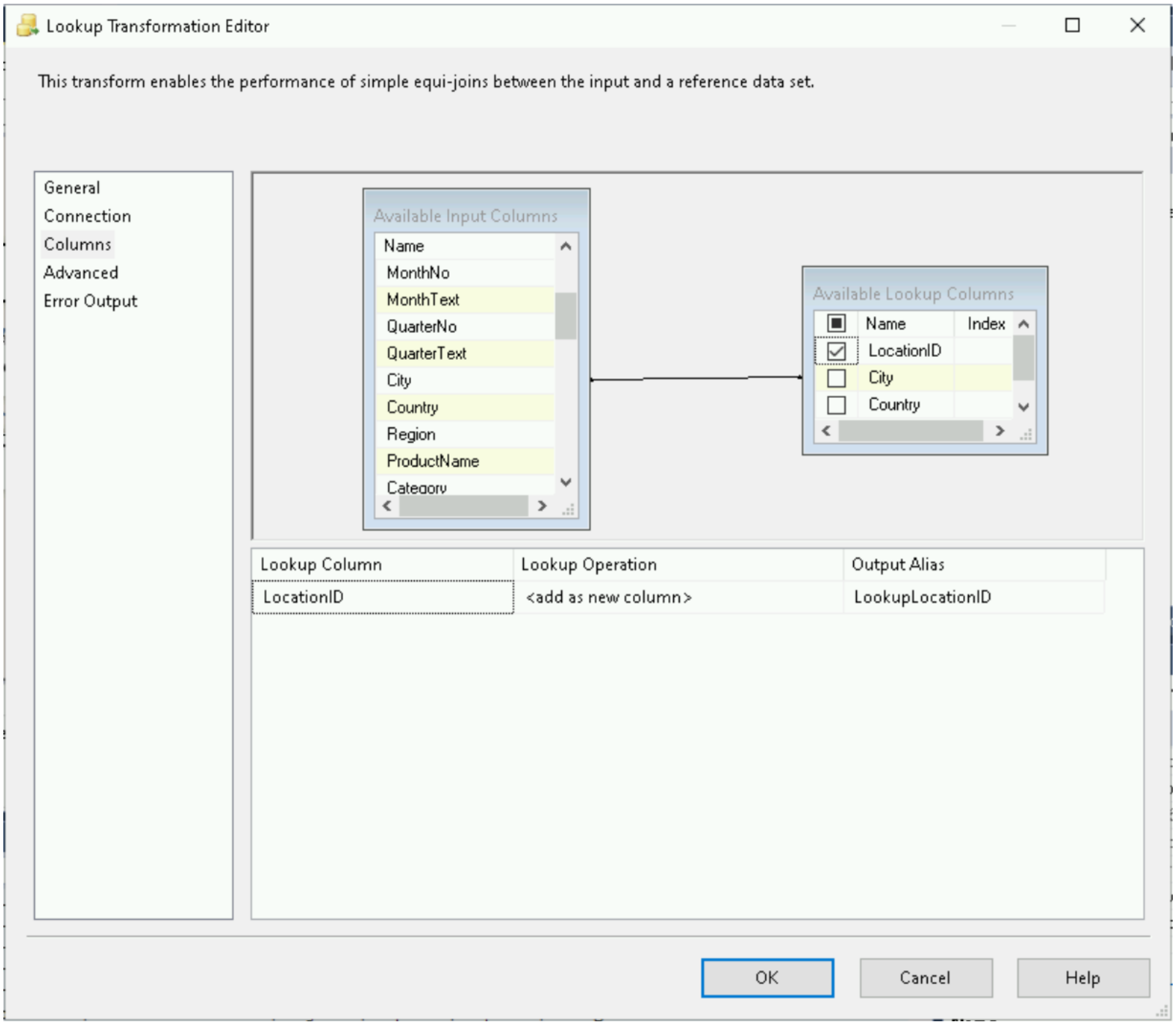
| Input Column | Destination Column |
|---|---|
| <ignore> | contactID |
| phone | phone |
| email | email |
| address | address |
| zipcode | zipcode |
| name | name |
| customerNumber | customerNumber |

**Table 8** ETL for FactSales - mapping

| Input Column | Destination Column |
|---|---|
| LookupTimeID | TimeID |
| LookupLocationID | LocationID |
| LookupProductID | ProductID |
| SalesCount | SalesCount |
| Cost | Cost |
| Profit | Profit |
| QuantitySold | QuantitySold |
| AverageShipmentDelay | AverageShipmentDelay |
| LookuporderSK | orderSK |
| LookupcontactID | contactID |
| orderRowNumber | orderRowNumber |

## Annex C

Lookup transformation in ETL for FactSales table

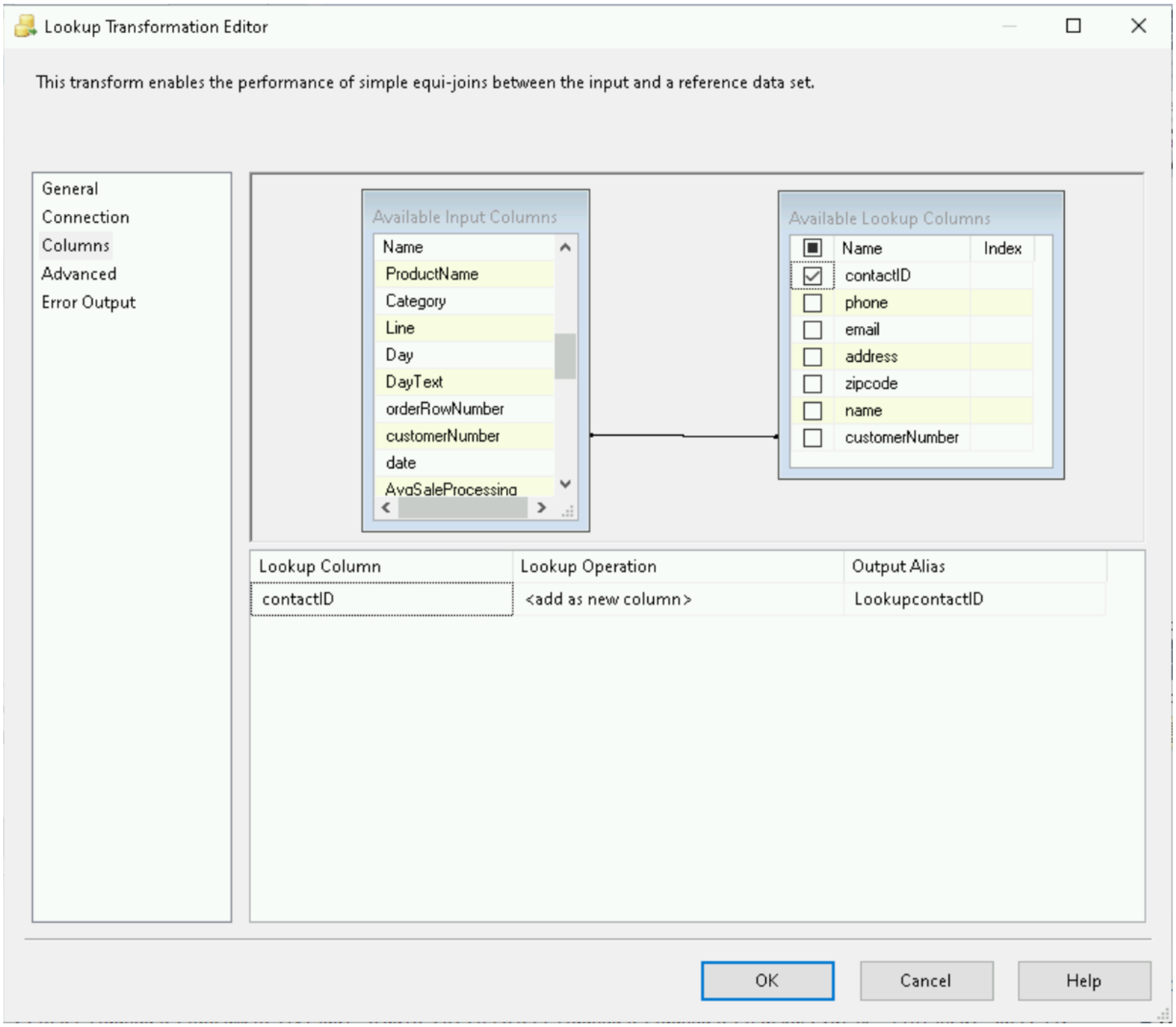


This transform enables the performance of simple equi-joins between the input and a reference data set.

☑ LocationID
□ City
□ Country

| Lookup Column | Lookup Operation | Output Alias |
|---|---|---|
| LocationID | <add as new column> | LookupLocationID |

Lookup LocationSK

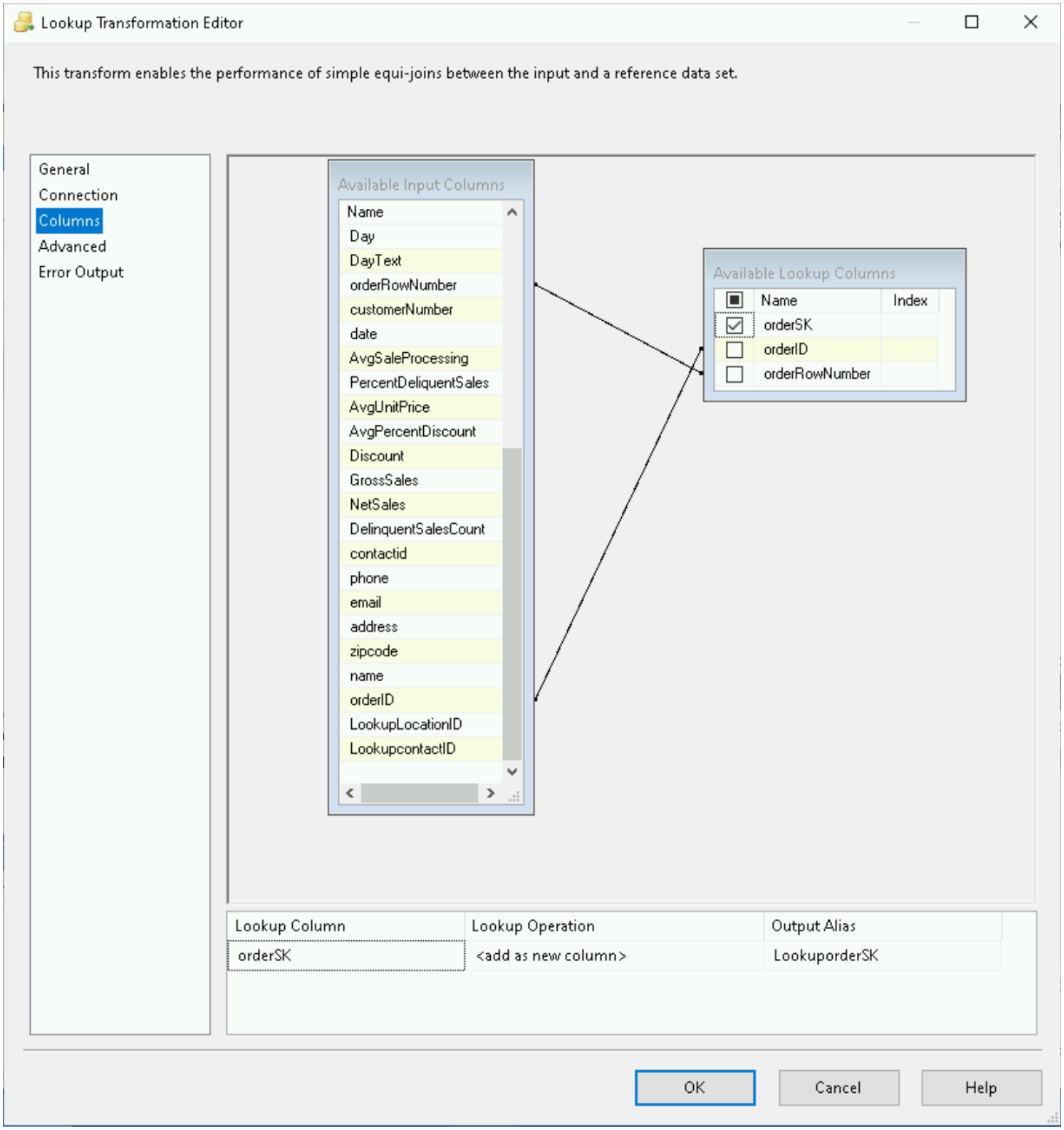


This transform enables the performance of simple equi-joins between the input and a reference data set.

☑ contactID
□ phone
□ email
□ address
□ zipcode
□ name
□ customerNumber

| Lookup Column | Lookup Operation | Output Alias |
|---|---|---|
| contactID | <add as new column> | LookupcontactID |

Lookup contactID



This transform enables the performance of simple equi-joins between the input and a reference data set.

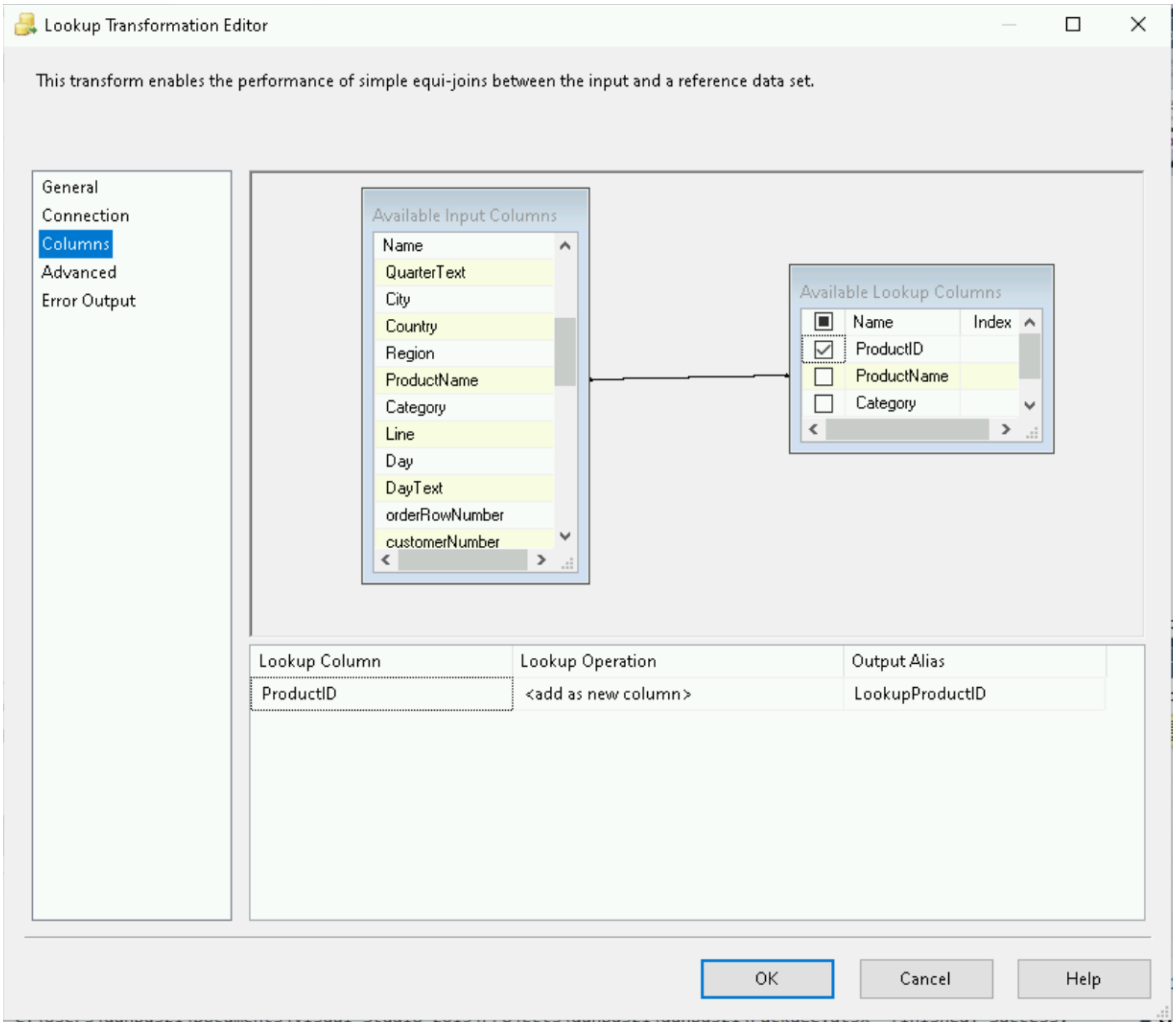
| Available Input Columns |
|---|
| Name |
| Day |
| DayText |
| orderRowNumber |
| customerNumber |
| date |
| AvgSaleProcessing |
| PercentDeliquentSales |
| AvgUnitPrice |
| AvgPercentDiscount |
| Discount |
| GrossSales |
| NetSales |
| DelinquentSalesCount |
| contactid |
| phone |
| email |
| address |
| zipcode |
| name |
| orderID |
| LookupLocationID |
| LookupcontactID |

Available Lookup Columns

| ■ | Name | Index |
|---|---|---|
| ☑ | orderSK | |
| □ | orderID | |
| □ | orderRowNumber | |

| Lookup Column | Lookup Operation | Output Alias |
|---|---|---|
| orderSK | <add as new column> | LookuporderSK |

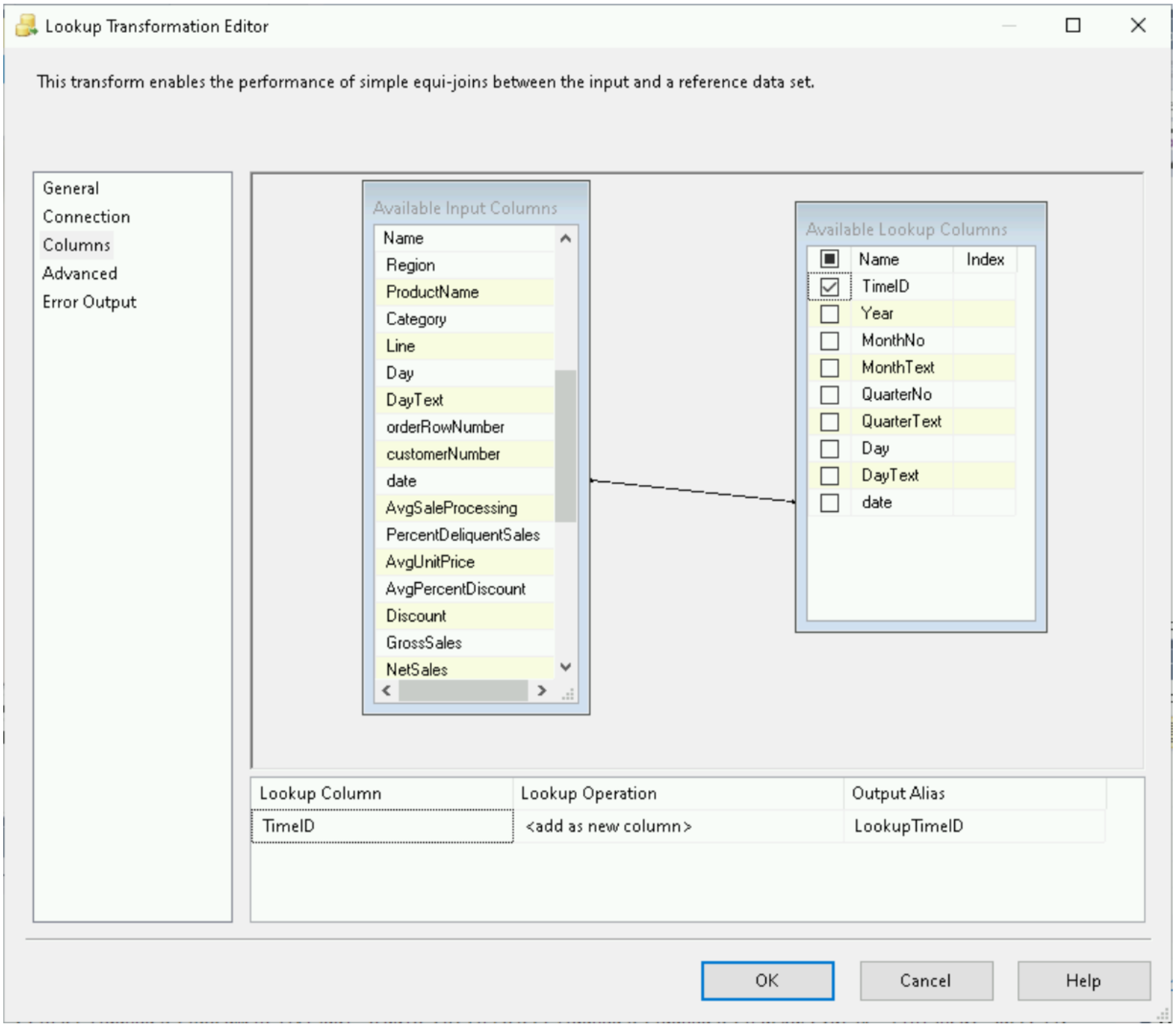
Lookup orderSK



This transform enables the performance of simple equi-joins between the input and a reference data set.

| Lookup Column | Lookup Operation | Output Alias |
|---|---|---|
| ProductID | <add as new column> | LookupProductID |

Lookup productSK



This transform enables the performance of simple equi-joins between the input and a reference data set.

Available Input Columns

| Name |
|---|
| Region |
| ProductName |
| Category |
| Line |
| Day |
| DayText |
| orderRowNumber |
| customerNumber |
| date |
| AvgSaleProcessing |
| PercentDeliquentSales |
| AvgUnitPrice |
| AvgPercentDiscount |
| Discount |
| GrossSales |
| NetSales |

Available Lookup Columns

| ■ | Name | Index |
|---|---|---|
| ☑ | TimeID | |
| □ | Year | |
| □ | MonthNo | |
| □ | MonthText | |
| □ | QuarterNo | |
| □ | QuarterText | |
| □ | Day | |
| □ | DayText | |
| □ | date | |

| Lookup Column | Lookup Operation | Output Alias |
|---|---|---|
| TimeID | <add as new column> | LookupTimeID |

Lookup TimeSK