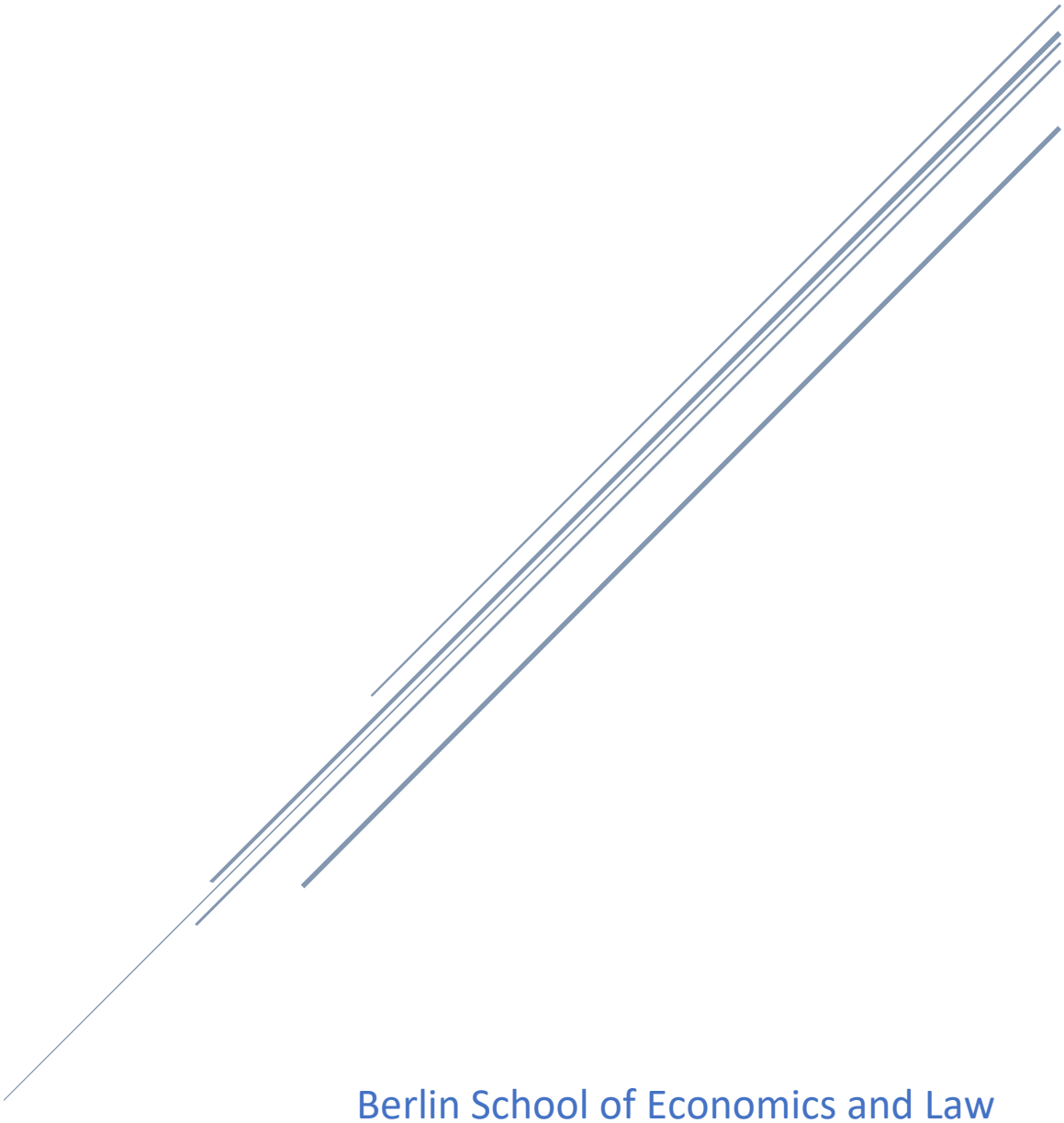
# SUPER-X CASE STUDY - SALES

Ayreen Japutri (1849354), Gesara Halili (1849118), Dustin Schwarz (1830240) & Elias Brummund (0519999)



Berlin School of Economics and Law
Data Warehousing

# Table of Content

# 1. Introduction

This project was dedicated to the Sales department of Super-X. The overall goal of the project was to give the Super-X management a better understanding of the performance of the Sales process. Therefore, a data mart was developed to help identify patterns in the data, which can aid the strategy development and decision making. Additionally, process mining was used to give a data-driven overview of the as-is process.

The major part of the report deals with the necessary steps which were conducted for building the data mart. As a guideline, the dimensional model design life cycle (DMDL) was followed. Starting with the analysis of the business requirements, we identified important questions which should be answered by our data mart. With the collection of the business requirements, we were able to analyze the data sources and identify the important tables from the raw data for which we conducted a data quality analysis with the help of Talend. After getting a fundamental understanding of the requirements and the data situation, we started with the conceptual design of the data mart. Therefore, the granularity and dimensions were derived from previous analyses and a ME/R diagram for the initial multi-dimensional design was created.

With the design set, we could start with the implementation by creating the structure of the Sales data mart with PowerArchitect, following our conceptual design and including the relevant information from the raw data. Since several data quality issues were identified in the previous analysis, a big part of the project was dedicated to the ETL process. Therefore, the required data were extracted from the operational database as well as the extra csv files and transformed before loading them to the data mart. For this step of the project, Pentaho was used.

After the implementation of the data mart was finished and the cleaned and prepared data were loaded, several dashboards were created to visualize the data and important KPIs for the Sales management of Super-X. For the visualization, Tableau and Microsoft Power BI were used as

tools. Based on our proof-of-concept implementations, an evaluation and comparison of the different technologies was conducted afterwards.

The second and minor part of the report covers the analysis of the processes with the help of process mining. Disco was used as a tool for undertaking this step. The process mining was based on the event log in the operational database. This part is an addition to the data mart since it offers different insights into the Sales performance and process.

Lastly, several business recommendations were derived from the findings of the previous parts. The suggestions help the Super-X management with the process improvement initiative for the Sales department.

## 2. Analysis of the business requirements

As a first phase of the project, the business requirements were identified. These are setting the base for designing the dimensional model and to think about important questions which should be answered. In *Table 1* we listed and evaluated all business requirements which are considered in the data mart design.

**Table 1: Business requirements**

| No. | Business Requirement | Importance | High Level Entities | Measures |
|---|---|---|---|---|
| 1 | What is the monthly/yearly number of orders per retailer? | High | Retailer, Month/Year | Number of orders |
| 2 | What is the average employees age? | Low | Employees | Average age |
| 3 | What is the most common activity done by employees per month? How long does the activity take in average? | High | Activity, Month | Maximum value of the sum of activities done. |

| | | | | |
|---|---|---|---|---|
| | | | | Average duration of activities. |
| 4 | Which are the top 5 retailers that the Sales department must call the most to remind them for the order list? | Medium | Activity, Retailers | Sum of reminder calls |
| 5 | Which are the top ten materials according to revenue? | Low | Materials | Price |
| 6 | Which is the biggest category of retailers that the company is working for? How many orders per retailer were placed? | Medium | Retailers | Sum of retailers per category, Order quantity |
| 7 | What are the sales for each sales employee? Who are top and bottom five employees? (excluding cancelled orders) | Medium | Employees | Number of orders per sales employee |
| 8 | What are the sales of all materials for each month? | High | Material, Weekday, Month | Sales quantity |
| 9 | How much revenue was made each year? How much revenue was created with OEM products vs own products? | High | Material, Year | Sales revenue |

| | | | | |
|---|---|---|---|---|
| 10 | What is the sales volume by location for each month/year? | High | Order, Retailer, Time | Number or order |
| 11 | What is the monthly sales growth rate compared to last year? | Medium | Order, Time | Sales revenue |
| 12 | What is the average quantity of material per order? | Low | Order, Product | Average number of materials |
| 13 | What is the cancellation rate per month and year? Which are the top ten retailers with the highest cancellation rate? | High | Order, Retailer | Cancellation rate (canceled orders / total orders) |
| 14 | How much is the sales loss per month? | Medium | Order, Retailer, Month | Revenue of cancelled orders |

After analyzing the business requirements, we filtered down the entities and measures in an information package template (*see Table 2)*. This is a way to derive potential future dimensions.

**Table 2: Information Package**

| **Entities (potential future dimensions)** | **Hierarchies in entity (potential future dimensions)** |
|---|---|
| Material/Product | Type -> Name |
| Retailer | Category -> Name<br>Country -> Region |
| Employees | Employee -> Department |
| Date | Year -> Quarter -> Month -> Day |
| Order | Order Status -> Order-ID |

| | |
|---|---|
| **Measures (Key Performance Indicators) (potential future measures in fact table)** | Sales revenue, Age, Price, Sum of retailers per category, Sales quantity, Order quantity per sales employee, Revenue per material, Sales quantity per material, Average number of materials per order, Order quantity per retailer, Cancellation rate, Revenue losses due to cancelled orders |

The tables above only contain the requirements which we included into the data mart. Despite these, we gathered more ideas which were not considered any further. All additional ideas are listed in the following:

- How much does the forecasted quantity differ from the actual ordered quantity each month, for each retailer, for each material?
- What is the additional ordered quantity during the month, for each retailer and for each material?
- Which are the top 5 materials with the highest change request during the month?
- How much of each material/product was ordered by the retailers each month/year?
- What is the cross-sell rate?
- What is the margin for each OEM material? How much profit for each OEM product is generated per year?
- Retailer churn rate – How many retailers does the company loose?

After analyzing the given data and our possibilities for the data mart, we decided to not include these extra ideas for reasons like the lack of needed information/data or because the project scope would have been expanded too much.

# 3. Analysis of the relevant data sources

Data profiling is the process of examining the data available in different data sources and collecting statistics and information about this data (Talend, 2017). It helps to assess the quality level of the data according to defined goals. Ignoring poor data qualities can lead to poor results. That is why it is important to first know the data quality to be able to deal with it and find ways to fix any problems. For this process Talend was be used. After coming up with the most important business requirement and checking the Super-X OLTP ER-Diagram, the following tables were identified as useful: materials, order items, orders, employees, retailers, event logs, inventories and bill of materials. It was also decided to consider the extra orders from the csv files, to provide more accurate and informative results.

## 3.1. Overall structural analysis

| Schema | #rows | #tables | #rows/table | #views | #rows/view | #keys | #indexes |
|---|---|---|---|---|---|---|---|
| public | 669851 | 38 | 17627.66 | 0 | NaN | 37 | 86 |

| Table | #rows | #keys | #indexes |
|---|---|---|---|
| purchase_orders | 4646 | 1 | 3 |
| retailers | 105 | 1 | 1 |
| schema_migrations | 0 | 0 | 1 |
| shipping_items | 72831 | 1 | 3 |
| shippings | 6429 | 1 | 4 |
| sim_clocks | 1 | 1 | 1 |
| sim_event_types | 0 | 1 | 1 |
| sim_events | 0 | 1 | 2 |

**Figure 1: Structural analysis output**

The schema overview analysis is especially useful to get a quick overview of the content in our schema. Here we can see that in total, we have 669,851 rows of 38 tables and 0 views. Only 37 out of 38 tables have keys. From the detailed table, we found out that *schema_migrations* is the only table that does not contain any key.

## 3.2. Retailers table analysis

**Retailer name matching analysis**

The matching analysis can be utilized to find the estimated number of groups of similar data or to find out about duplicate data on a table or a column set basis. Here, we used this method to analyze retailers name with the exact fax number as the matching function.

Match Rule 2 ×

| Match Key Name | Input Column | Matching Function | Custom Matcher | Tokenized measure |
|---|---|---|---|---|
| fax | fax | Exact | | No |
| name | name | Soundex | | No |

**Figure 2: Retailer name matching analysis rules**

With the analysis, we found out that several retailers have the same or similar name as shown in *Figure 3*.

| id | name | fax |
|---|---|---|
| 18 | Frami Inc | 803-913-0682 |
| 102 | FramicIn | 803-913-0682 |
| 83 | Boer, Kok and Dam | 0658685192 |
| 101 | Boer, Kok anD dam | 0658685192 |
| 34 | Rath, Schuppe and Runte | 586-541-3422 x9594 |
| 104 | Rath, Schuppe and Runte | 586-541-3422 x9594 |
| 4 | Prosacco-Kub | (999) 999-999 |
| 105 | Puosacco-Krb | (999) 999-999 |
| 98 | Charles et Lucas | 0322176609 |
| 103 | Charles et Lucas | 0322176609 |

**Figure 3: Duplicate retailers**

After comparing the duplicates shown in Figure 3: Duplicate retailers*Figure 3* with the order table, we found out that all the assumed duplicate retailers exist and order since 2010 to 2017. Therefore, both retailers were kept.

| | id | state | retailer_id | employee_id | timestamp |
|---|---|---|---|---|---|
| 1 | 58 | Canceled | 102 | 51 | 2010-02-26 09:00:00 |
| 2 | 179 | Canceled | 102 | 39 | 2010-03-26 09:00:00 |
| 3 | 69 | Canceled | 18 | 48 | 2010-02-26 09:00:00 |
| 4 | 271 | Shipped | 102 | 48 | 2010-04-27 09:00:00 |
| 5 | 280 | Shipped | 18 | 33 | 2010-04-27 09:00:00 |
| 6 | 169 | Canceled | 18 | 37 | 2010-03-26 09:00:00 |
| 7 | 377 | Canceled | 18 | 114 | 2010-05-27 09:00:00 |
| 8 | 385 | Shipped | 102 | 88 | 2010-04-27 09:00:00 |

**Figure 4: Orders from duplicate retailers**

### Retailer email pattern matching

The pattern matching analysis regarding the retailer email addresses enables us to profile the email with the correct pattern.

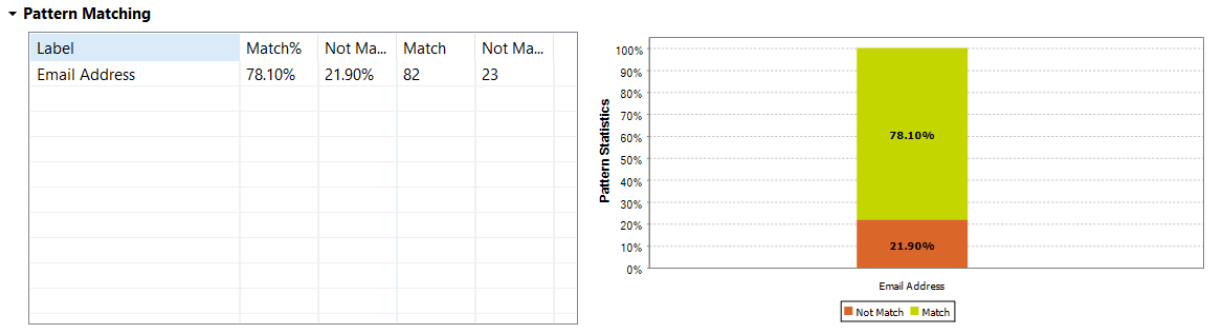



**Figure 5: Result of retailer email pattern matching**

We found out that 21.9% of the retailer email addresses are not matching to the email format, as it can be seen in *Figure 6*.

| email |
|---|
| van_janssen_thijs@wal nl |
| nicol_sr_ulloa_s_lebr_n@alvarez.e |
| sch eler.loyce@mann.org |
| rau.demon @johns.info |
| mme_baro _romane@gonzalez.com |
| eva.brin @vriesbrouwer.org |
| berge_beth@f hey.org |
| ebony.schaefer@harveytremblay com |
| prof_juliette_colin@ irard.fr |
| pri to.lucia.almonte@vega.es |
| jewel.osinski@rayno kirlin.info |
| elna_stehr@stantonn colas.biz |
| jermaine.rutherford@wit ing.ca |
| mazza_adriano@milan .org |
| mathis.m. eclerc@robert.org |
| niak.oleg.le@decfalkows i.pl |
| kobe. oodwin@towne.co.uk |
| urbansky_la ra@hommel.com |
| mclau hlin_derick@ullrichwest.com |
| wiza.darian@s orer.name |
| raik.birkemeyer@g eithanner.org |
| bradtke.sydnie@bergstr m.biz |
| olivier.alexis@richa d.net |

**Figure 6: Retailer emails with wrong pattern**

In a real-life case, we should contact each retailer to get their correct email address, especially if it is the preferred way to communicate with each other. In our case we are going to ignore this problem as we have no way to find the correct email addresses.

## Retailer category frequency

**Value Frequency**

| Value | Count | % |
|---|---|---|
| offline | 32 | 30.48% |
| offline retailer | 29 | 27.62% |
| online | 24 | 22.86% |
| e-shop | 20 | 19.05% |

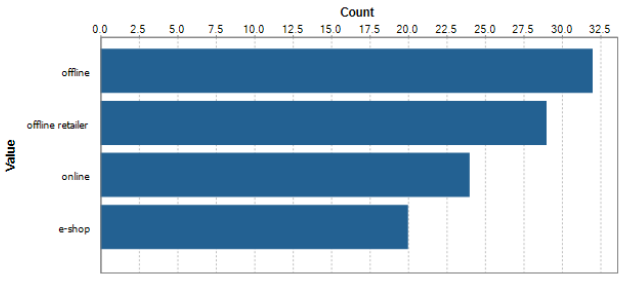



**Figure 7: Result of retailer category value frequency analysis**

Applying the pattern frequency analysis for the retailer category, helped us to find related categories which should be merged to one. The analysis shows that there are four most frequent retailer categories which have a similar function. Onwards, we squished these categories into offline (containing ‘offline’ and ‘offline retailer’) and online (containing ‘online’ and ‘e-shop’).

### Retailer phone statistical analysis

**Column: retailers.phone**

**Simple Statistics**

| Label | Count | % |
|---|---|---|
| Row Count | 105 | 100.00% |
| Null Count | 0 | 0.00% |
| Distinct Count | 87 | 82.86% |
| Unique Count | 86 | 81.90% |
| Duplicate Count | 1 | 0.95% |
| Blank Count | 0 | 0.00% |

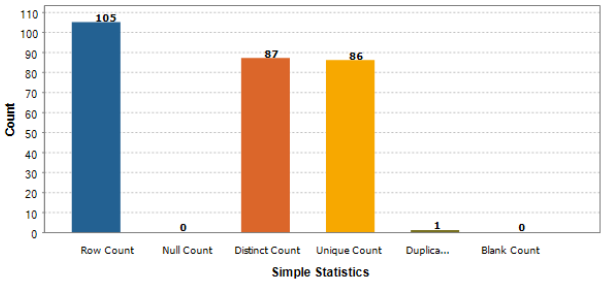



**Figure 8: Result of retailer phone statistical analysis**

Applying the summary statistical analysis for the retailer phone and fax numbers, helped us to find one duplicating phone number which is (999) 999-999 for 19 retailers. Since this seems to be a placeholder for a missing value, it should be set to null. An overview of the retailers with this number can be found in the following figure.

| id | name | phone |
|---|---|---|
| 5 | Dekker V.O.F. | (999) 999-999 |
| 13 | Hardy UG | (999) 999-999 |
| 21 | Wiśniewski, Wysocki and Grzegorczyk | (999) 999-999 |
| 23 | Sanford and Sons | (999) 999-999 |
| 34 | Rath, Schuppe and Runte | (999) 999-999 |
| 35 | Gabler OHG | (999) 999-999 |
| 36 | Kozey Group | (999) 999-999 |
| 37 | Krohn Gruppe | (999) 999-999 |
| 39 | Stürmer, Grotke und Gutowicz | (999) 999-999 |
| 50 | Daugherty-Effertz | (999) 999-999 |
| 51 | Villa-Ferretti SPA | (999) 999-999 |
| 58 | Weimer, Scheuring und Schönball | (999) 999-999 |
| 62 | Schuppe, Feil and Flatley | (999) 999-999 |
| 63 | Ferretti, Monti e Fabbri e figli | (999) 999-999 |
| 67 | Bieler-Ochs | (999) 999-999 |
| 68 | Frączek Group | (999) 999-999 |
| 69 | Lehner Group | (999) 999-999 |
| 77 | Kunde, Welch and Pfeffer | (999) 999-999 |
| 98 | Charles et Lucas | (999) 999-999 |

**Figure 9: List of retailers with phone number (999) 999-999**

### Retailer phone frequency

**Pattern Frequency**

| Value | Count | % |
|---|---|---|
| () - | 20 | 19.05% |
| --- | 13 | 12.38% |
| Empty field | 12 | 11.43% |
| | 9 | 8.57% |
| -- a | 9 | 8.57% |
| + | 7 | 6.67% |
| +-- | 7 | 6.67% |
| -- | 6 | 5.71% |
| 0 | 6 | 5.71% |
| .. a | 5 | 4.76% |

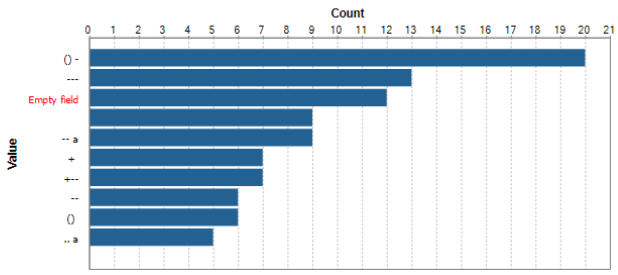



**Figure 10: Retailer phone pattern frequency analysis result**

The advanced analysis result of the retailer phone summary statistic also shows that there are several formats of a retailer's phone number in the database. In the future, Super-X should agree on one format for the phone and fax numbers to have a clean data mart.

### Retailer contact person statistic

**Column: retailers.contactperson**

**Simple Statistics**

| Label | Count | % |
|---|---|---|
| Row Count | 105 | 100.00% |
| Null Count | 4 | 3.81% |
| Distinct Count | 102 | 97.14% |
| Unique Count | 101 | 96.19% |
| Duplicate Count | 1 | 0.95% |
| Blank Count | 0 | 0.00% |

**Figure 11: Result of retailer contact person statistical analysis**

After applying the summary statistical analysis for the retailer contact person column, the result shows us that there are four null values. Since the contact person is not important for our transaction, we ignored those null values.

**Retailer address analysis**

Another problem we found is that the entire retailer address is concatenated in one column, including: street, ZIP code and district, region, country.

| address |
|---|
| Kevinstraat 279c, 3804 IP Daandam, Groningen, Netherlands |
| Travesía Luis Miguel Ramírez, 9, 93087 Linares, Comunidad de Madrid, Spain |
| 2446 Odie Ridge, SH1 7JL North Max, Northern Ireland, United Kingdom |
| 60106 Amara Throughway, 84189-9553 Bashirianshire, Tennessee, USA |
| Broeklaan 199 II, 4256 BA Oost Emmakerk, Limburg, Netherlands |
| Lado María Soledad s/n., 26149 El Ejido, Canarias, Spain |
| 730 Johnson Terrace, 72020-6534 Port Korbin, Wyoming, U.S.A. |

**Figure 12: Retailer address list**

This kind of address will limit the future analysis. Therefore, the street, district, region and country should be separated into four different columns. A detailed explanation on how we separated this column is part of chapter 5.1.

## 3.3. Order items table analysis

**Column analysis**

After conducting a column analysis for all columns within the order items table, a data quality problem was detected for the currency column.

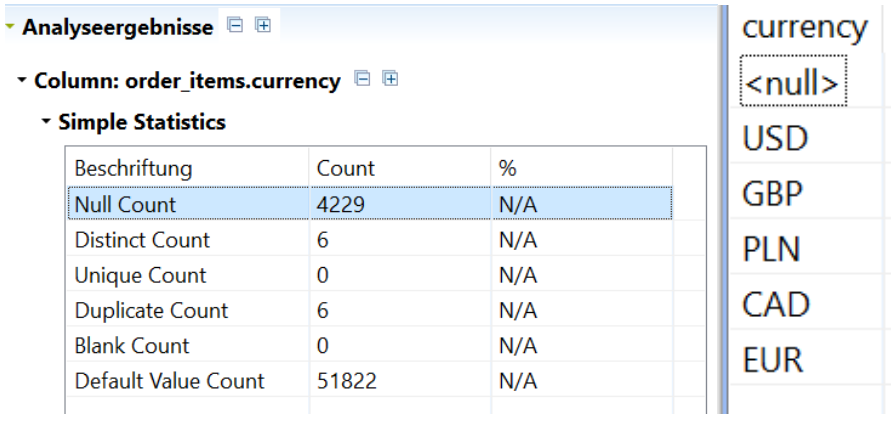



**Figure 13: Column analysis of currency**

The statistics show that there are 4.229 missing values within the currency column which causes a problem when we want to calculate the revenue per order in the fact table of the data mart. This problem should be fixed within the ETL process. The statistics also show that there are five different currencies (excluding the Null values). This is also important to know for calculating the total revenue in one specific currency since we need to consider the exchange rates for all these different currencies.

### Cross table redundancy analysis

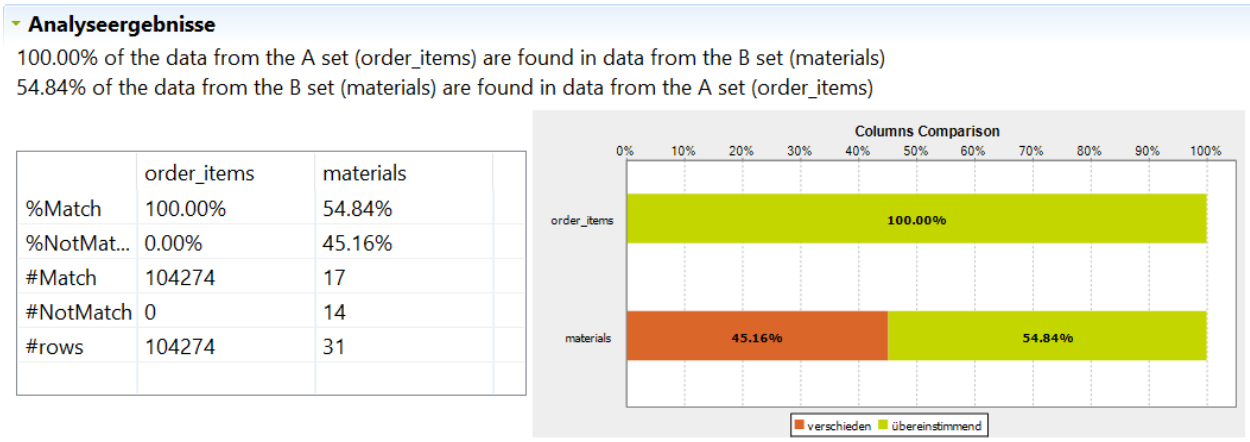



**Figure 14: Cross table redundancy analysis of material ids**

The cross table redundancy analysis was conducted for the columns order_id and material_id to check for any missing linkages. The figure shows that all material_id's within the order_items table are present in the materials table but not vice versa. 14 of the materials are not linked to the order_items table which means that these materials were not ordered at all from the retailers. Taking a closer look, these materials are exclusively raw materials or semi-finished products which explains this situation.

Despite the previously mentioned findings, no other data quality problems like missing data, invalid values or missing linkages have been found within the order_items table.

## 3.4. Orders table analysis

Firstly, we conducted a column analysis for the orders table. It showed that this table does not contain any missing or invalid values. An interesting information is that 9.111 orders were made by 105 different retailers. Apparently, 28 employees are working for Sales and were responsible for the orders.

As a next step a cross table redundancy analysis was executed for the employee_id and retailer_id column. No missing linkages between these columns and the corresponding tables were found. All employees that were not linked to the orders table do not work in the Sales department which indicates that there is no data quality problem.

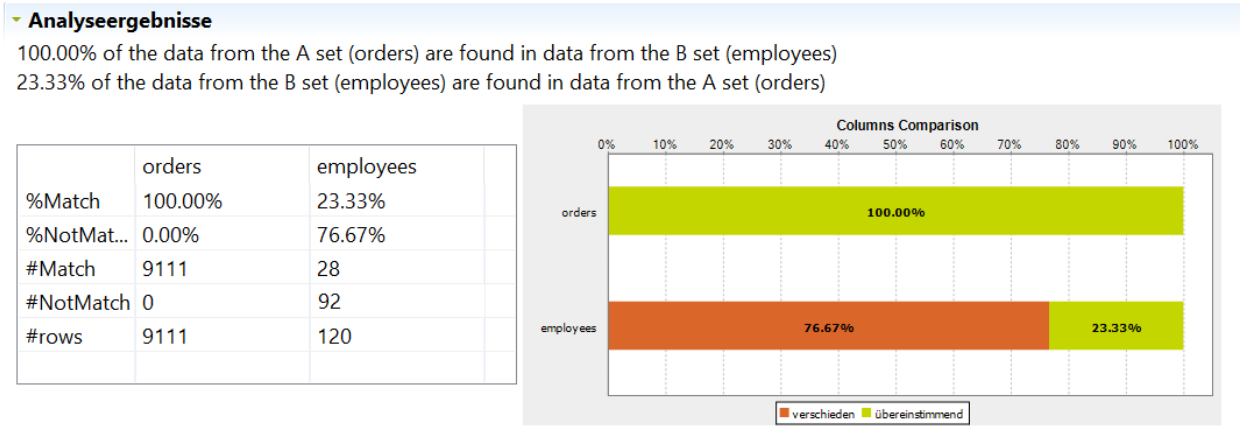



**Figure 15: Cross table redundancy analysis of employees in the order items table**

## 3.5. Event logs table analysis

**Column Analysis**

Even though, the event_logs table will not be part of the data mart, the data quality was also analyzed for this table as it will be used for the process mining. Some issues with the table were found during the column analysis.

**Column: event_logs.supplier_id**

**Simple Statistics**

| Beschriftung | Count | % |
|---|---|---|
| Row Count | 55528 | 100.00% |
| Null Count | 55528 | 100.00% |
| Distinct Count | 1 | 1.801E-3% |
| Unique Count | 0 | 0.00% |
| Duplicate Count | 1 | 1.801E-3% |

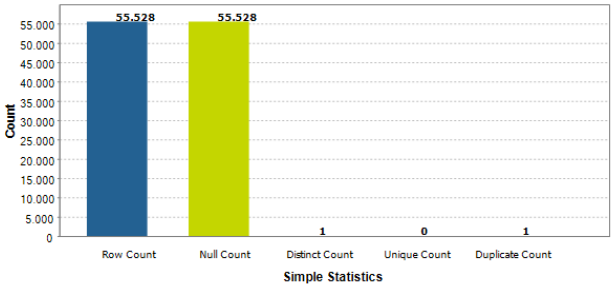



**Figure 16: Column analysis of the supplier ids**

The first finding refers to the column supplier_id which does not contain any values. This leads to the question why this column is in the table since it does not fulfill any purpose. But since the suppliers are not important for the Sales department, this problem can be ignored for this project.

**Column: event_logs.employee_id**

**Simple Statistics**

| Beschriftung | Count | % |
|---|---|---|
| Row Count | 55528 | 100.00% |
| Null Count | 9047 | 16.29% |
| Distinct Count | 29 | 0.05% |
| Unique Count | 0 | 0.00% |
| Duplicate Count | 29 | 0.05% |

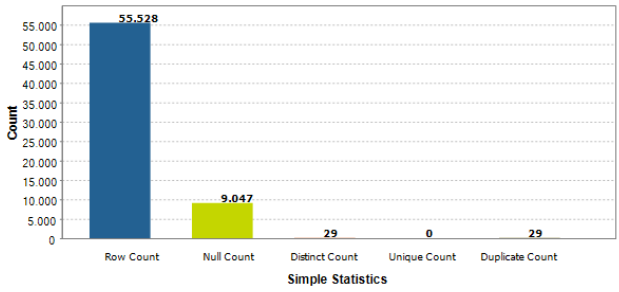



**Figure 17: Column analysis of the employee ids**

Another finding is that in 9.047 rows no employee_id was entered which means that we don't know the person who executed the activity. After taking a closer look into the rows with a Null value, we found out that the employee_id is only missing for activities regarding the Logistics department or external employees. Therefore, it is not relevant for the Sales department.

**Functional Dependency Analysis**

To check if there any cases where an activity was linked to a wrong department, a functional dependency analysis was conducted. The result shows that there are cases where different departments were linked to the same activity (see Figure 18). This is the case for the activity of undertaking a reminder call to Sales which was linked to Procurement and Production Planning. This seems to be problematic at the first sight but regarding the process descriptions from the BPM course, this activity is actually done by these two departments and therefore, it is not a data quality issue.

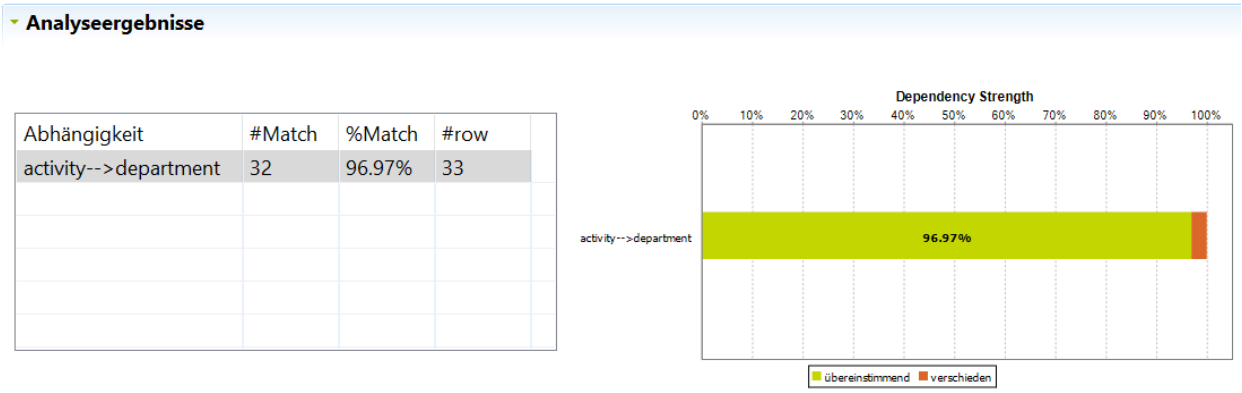



**Figure 18: Functional dependency analysis of activity and department**

**Cross Table Redundancy Analysis**

Additionally, a cross table redundancy analysis was done for this table. While checking for cross table redundancies between the column employee_id and the employee table, we noticed one thing. Only 23.33% of all employee ids are found in the event_logs table which means that the activities are only done by some of the employees. The 16.29% of the values from the employee_id column that cannot be found in the employee table are referring to Null values.

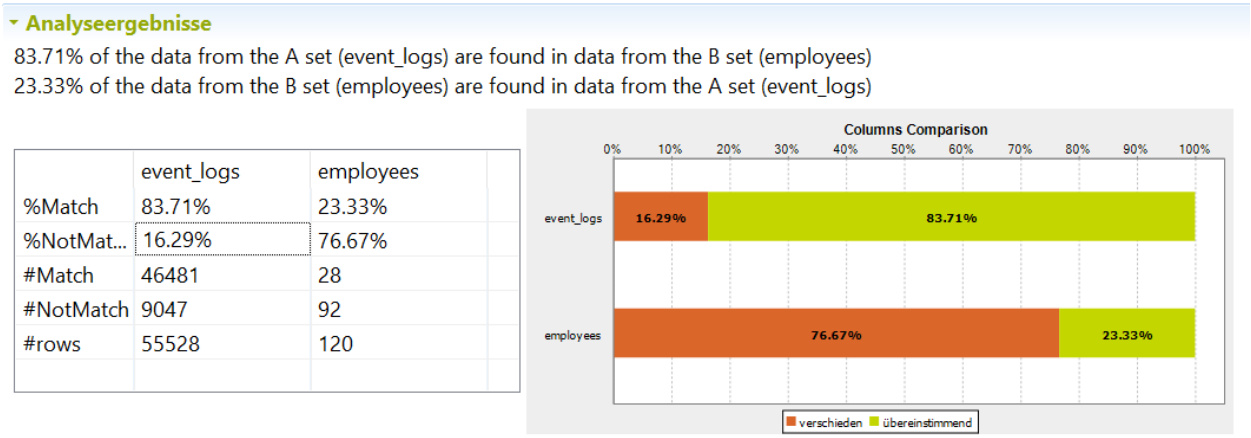



**Figure 19: Cross table redundancy analysis of employees in the event logs table**

## 3.6. Extra orders table analysis

Extra orders are collected from another system and are saved within csv files. Firstly, we uploaded these files to a table in the database to have a better view on how the data look like. The table name is create_table_from_csv.

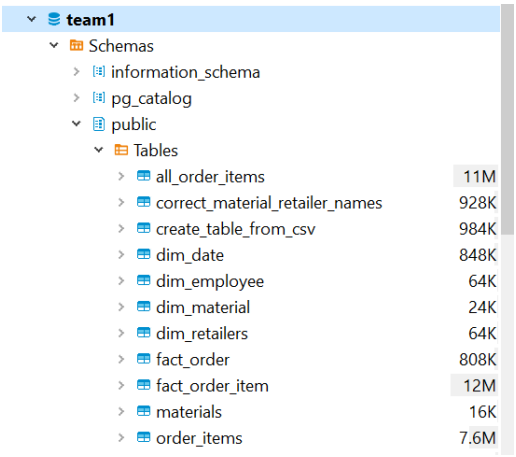



**Figure 20: Creating table from csv files**

While analyzing these tables we noticed a specific problem. Retailer and material names are sometimes written in a wrong way, which can be a human error during typing (misspelling values).

team1 | dim_date | public | fact_order_item | *<team1> Script-38

```
select   m.id as material_id, material as material_name, r.id, eo.retailer,
                eo.quantity, eo.price, eo.currency,  eo."Timestamp" as timestamp
from create_table_from_csv eo

left join materials m on m.name = material
left join retailers r on r."name" = eo.retailer
```

materials(+) 1

select m.id as material_id, material as r | Enter a SQL expression to filter results (use Ctrl+Space)

| | material_id | material_name | id | retailer | quantity | price | currency |
|---|---|---|---|---|---|---|---|
| 1 | [NULL] | BIPM ExpertsrLogo Sticke s | 75 | Murphy, Weber and Runolfsdottir | 2 | 0.79 | GBP |
| 2 | [NULL] | Boister Beast Logo Stockers | 52 | Daniel-Marvin | 14 | 0.4 | GBP |
| 3 | [NULL] | Booster Beast LogS otickers | 99 | Tillack, Schwarz und Lichtenfeld | 4 | 0.5 | EUR |
| 4 | [NULL] | SuperX- Booster Beast | [NULL] | Brown-Roberts | 2 | 84.87 | GBP |
| 5 | [NULL] | Su-erpX Buggy Champ | 15 | Schouten, Ven and Bos | 1 | 137.52 | EUR |
| 6 | [NULL] | Su-erpX Buggy Champ | [NULL] | Villa-Ftrretei SPA | 7 | 137.52 | EUR |
| 7 | [NULL] | Syper-X Buggu Champ | [NULL] | Abreu y VÃ¡zquez | 4 | 137.52 | EUR |
| 8 | [NULL] | Syper-X Buggu Champ | 38 | Tamayo y Tijerina | 4 | 137.52 | EUR |
| 9 | [NULL] | Syper-X Buggu Champ | 64 | Predovic-Bernhard | 11 | 151.27 | USD |
| 10 | [NULL] | Supex-X BIPM Erpert Racer | [NULL] | Honz, Schima rnd Effleu | 4 | 54.34 | EUR |
| 11 | [NULL] | Supex-X BIPM Erpert Racer | [NULL] | Honz, Schima rnd Effleu | 4 | 54.34 | EUR |
| 12 | [NULL] | Remote Controller  MHz1 | 29 | Madubuko-Clarius | 6 | 11.89 | EUR |
| 13 | [NULL] | Remote Controller 2-ChaHnel 2Mnz | 34 | Rath, Schuppe and Runte | 4 | 25.85 | USD |
| 14 | [NULL] | Remote Controller 2-ChaHnel 2Mnz | 104 | Rath, Schuppe and Runte | 4 | 25.85 | USD |
| 15 | [NULL] | BIPM ExpertseLogo Stick rs | 64 | Predovic-Bernhard | 7 | 1.09 | USD |
| 16 | [NULL] | konster TrucM Logo Stickers | 34 | Rath, Schuppe and Runte | 2 | 0.55 | USD |
| 17 | [NULL] | konster TrucM Logo Stickers | 104 | Rath, Schuppe and Runte | 2 | 0.55 | USD |
| 18 | [NULL] | Super-X BIPM Ex ertpRacer | 87 | Bergstrom, Robel and Wisozk | 5 | 59.77 | USD |

**Figure 21: Identifying misspelled names for retailers and materials**

Using left join we can identify these names that are not written in the right way. Also, for each of these extra orders, we do not have order_id. This means that we cannot identify to which order these belong to. That makes it complicated to integrate them with the other order items that we already have in the database. The solution on how to fix these problems will be given in detail in the chapter 5.1. For a better understanding of the data, a data quality analysis was carried out using Talend which gave the following results.

**Column analysis**

The tables in the csv files have six columns in total: material, retailer, quantity, price, currency and timestamp. After doing a column analysis for each column of the new table create_table_from_csv, there is not any problem about data quality detected. The tables below give us an information about how the data look like.

**Column: create_table_from_csv.material**

**Simple Statistics**

| Label | Count | % |
|---|---|---|
| Row Count | 8568 | 100.00% |
| Null Count | 0 | 0.00% |
| Distinct Count | 652 | 7.61% |
| Unique Count | 503 | 5.87% |
| Duplicate Count | 149 | 1.74% |
| Blank Count | 0 | 0.00% |

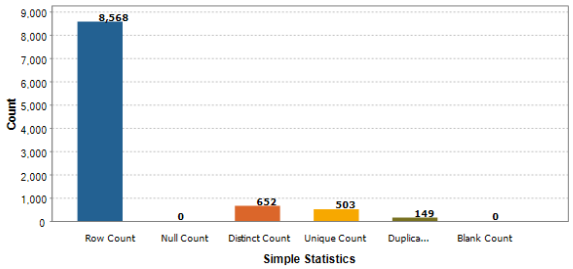



**Figure 22: Material column analysis for extra orders**

**Column: create_table_from_csv.retailer**

**Simple Statistics**

| Label | Count | % |
|---|---|---|
| Row Count | 8568 | 100.00% |
| Null Count | 0 | 0.00% |
| Distinct Count | 858 | 10.01% |
| Unique Count | 496 | 5.79% |
| Duplicate Count | 362 | 4.23% |
| Blank Count | 0 | 0.00% |

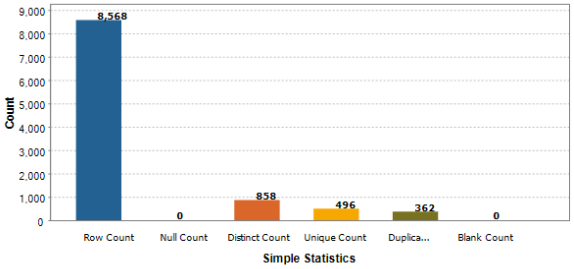



**Figure 23: Retailer column analysis for extra orders**

**Column: create_table_from_csv.quantity**

**Simple Statistics**

| Label | Count | % |
|---|---|---|
| Row Count | 8568 | 100.00% |
| Null Count | 0 | 0.00% |
| Distinct Count | 59 | 0.69% |
| Unique Count | 14 | 0.16% |
| Duplicate Count | 45 | 0.53% |

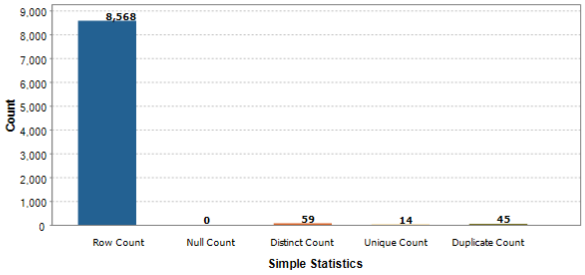



**Figure 24: Quantity column analysis for extra orders**

**Column: create_table_from_csv.price**

**Simple Statistics**

| Label | Count | % |
|---|---|---|
| Row Count | 8568 | 100.00% |
| Null Count | 0 | 0.00% |
| Distinct Count | 70 | 0.82% |
| Unique Count | 0 | 0.00% |
| Duplicate Count | 70 | 0.82% |

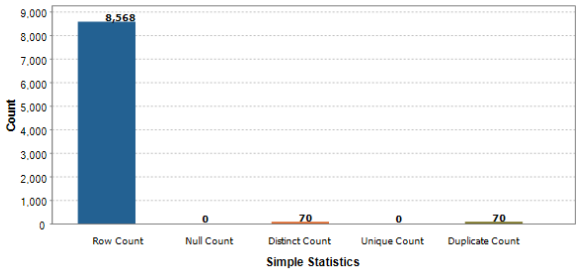



**Figure 25: Price column analysis for extra orders**

**Column: create_table_from_csv.currency**

**Simple Statistics**

| Label | Count | % |
|---|---|---|
| Row Count | 8568 | 100.00% |
| Null Count | 0 | 0.00% |
| Distinct Count | 5 | 0.06% |
| Unique Count | 0 | 0.00% |
| Duplicate Count | 5 | 0.06% |
| Blank Count | 0 | 0.00% |

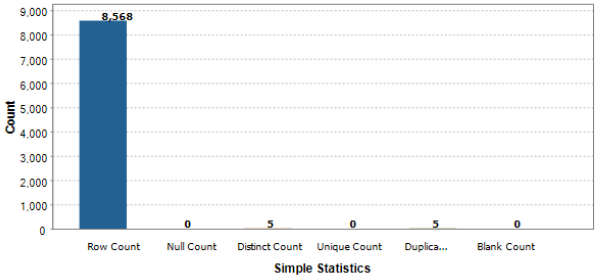



**Figure 26: Currency column analysis for extra orders**

It is clearly seen that there are no null values which means that we do not have to deal with this problem during the ETL process.

### Functional dependency analysis

In this case, considering the columns that this table has, we can check if the currency matches with the retailer in all cases. As it is known from data in the database, each retailer makes all the orders using the same currency, depending on the country this retailer is located.

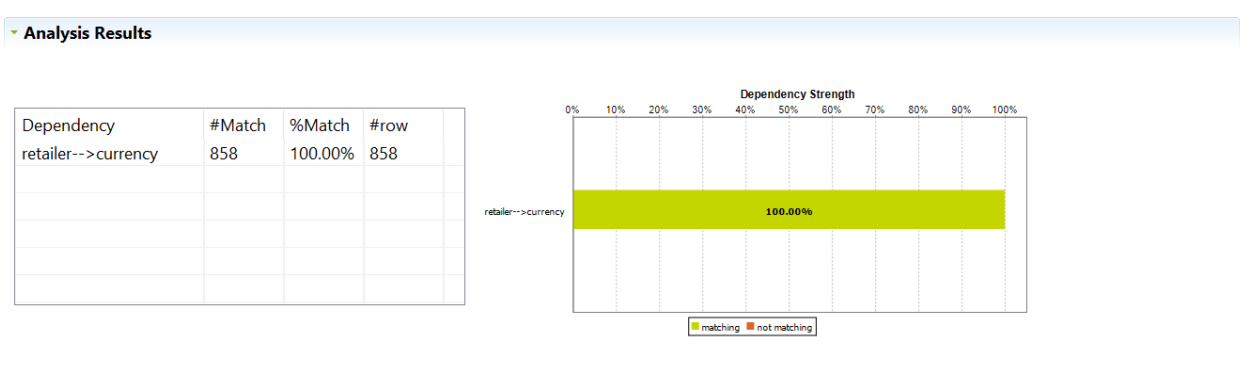



**Figure 27: Functional dependency analysis for retailer and currency columns**

As it can be seen, for each retailer we have exactly one currency. This is a good result and it means that the quality of data is good for this part, too.

### Cross Table Redundancy Analysis

The cross table redundancy analysis is conducted to see if the materials, retailers and currencies from the csv files are similar to the data which already exist in the database. Another interesting element to be checked is the timestamp.

Regarding the materials name, the results show that 9.3% of the materials are not found in the database.

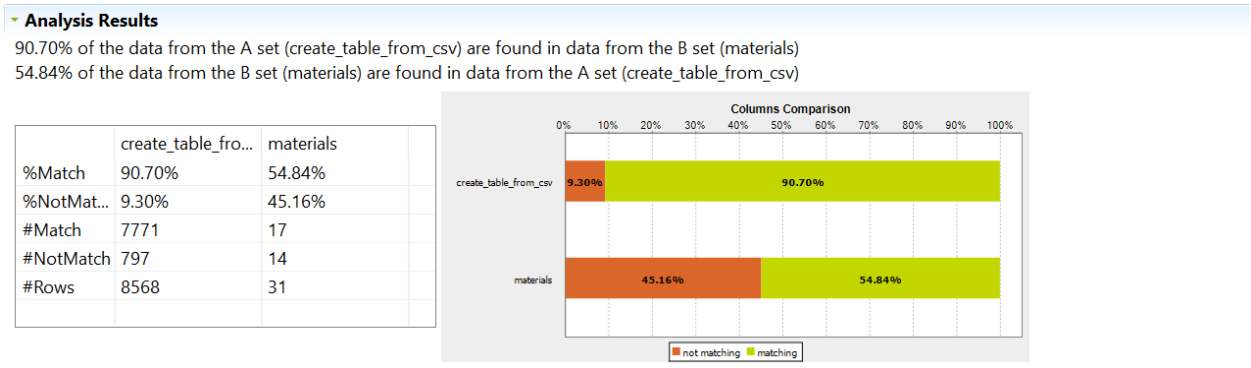



**Figure 28: Cross table redundancy analysis for material name**

The reason is not because there are new materials, but because some of the material names are written wrong in the csv files. There are exactly 797 material names which need to be corrected (see *Figure 28)*.

The same thing can be said for the retailers. In this case, there are 2.166 retailer names that cannot be found in the database. Again, the reason is a misspelling of the name (see *Figure 29)*.

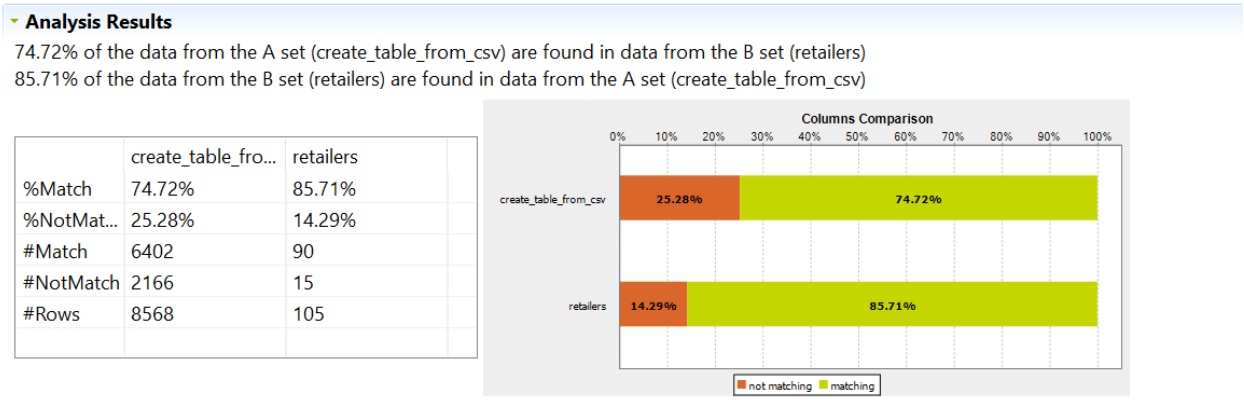



**Figure 29: Cross table redundancy for retailer name**

For the currency, the results show that all of them can be found in the database. So, there is no option for a misspelled currency name or a new currency.

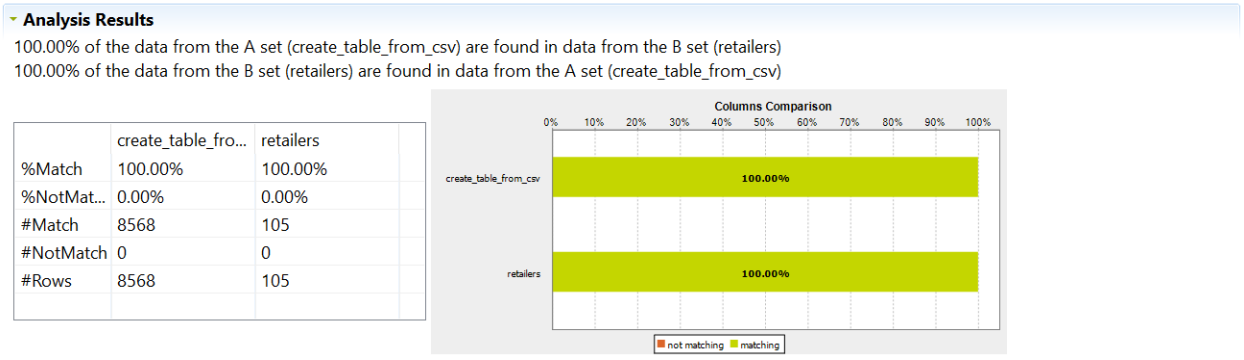



**Figure 30: Cross table redundancy for currency**

Last but not the least, we had a look at the timestamp. As we can see from *Figure 31*, 129 of the order times from the csv file do not match with the order times from the database. It can be derived that sometimes a retailer requires extra material some days after an order was placed.

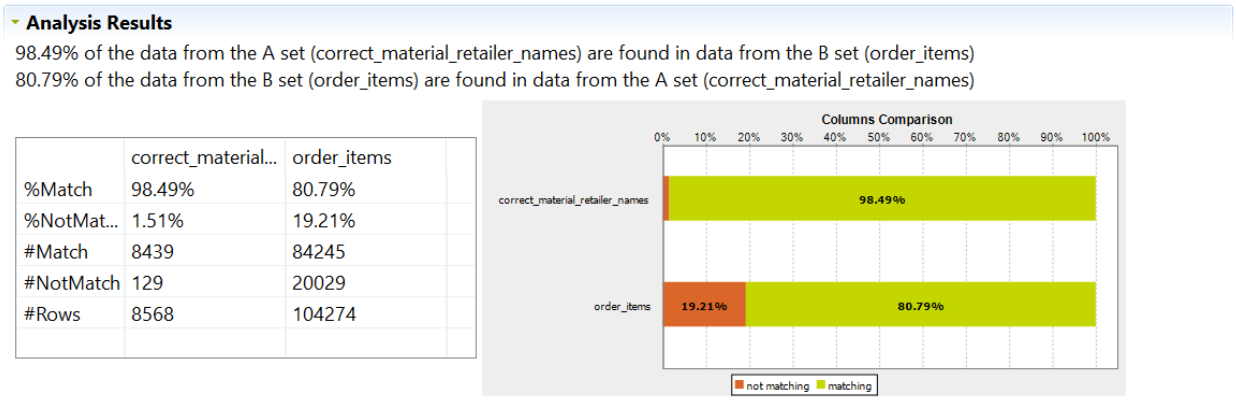



**Figure 31: Cross table redundancy for timestamp**

This happens in 1.51% of the cases. To understand it better, an example can be taken from the create_table_from_csv table. As shown in the *Figure 32*, "Cormier and Sons" placed their order on the 1st January 2010. But another item was ordered on 8th of January which would be the extra order.

| | material | retailer | quantity | price | currency | Timestamp |
|---|---|---|---|---|---|---|
| 1 | Super-X BIPM Expert Racer | Cormier and Sons | 3 | 59.77 | USD | 2010-01-01 09:00:00 |
| 2 | Ni-Cd Battery 12V 300mAh | Cormier and Sons | 3 | 2.75 | USD | 2010-01-01 09:00:00 |
| 3 | Tire 20 mm | Cormier and Sons | 6 | 1.2 | EUR | 2010-01-08 09:00:00 |

**Figure 32: Timestamp for retailer "Cormier and Sons" from csv files**

From Figure 33 we can see that all orders which were placed by the retailer for January 2010 are made on the 1st, too. The first option for this could be that the date in the csv file is wrong, but there is no way to prove it. Second case scenario is that Sales have a deadline until all orders can be made before proceeding to the next steps.

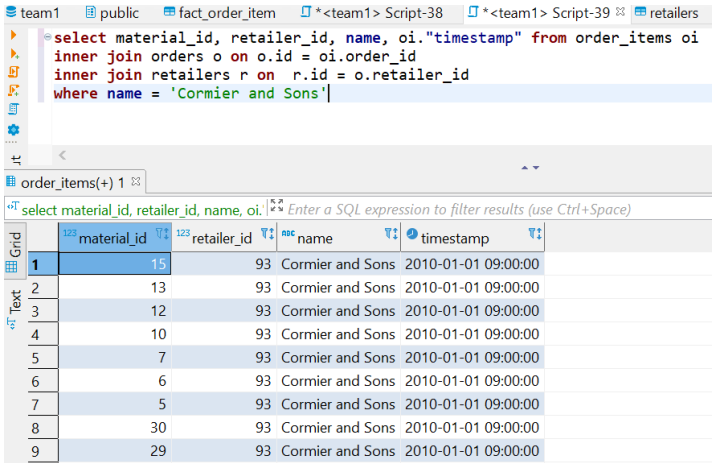



**Figure 33: Timestamp for retailer "Cormier and Sons" from database**

## 3.7. Employees table analysis

**Basic Column Analysis**

The first analysis performed on the employee table is a basic column analysis. On the screenshot below, it can be seen that 100% of the employees are apparently male.

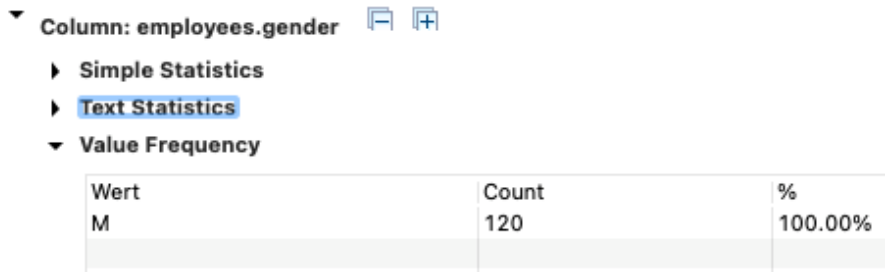
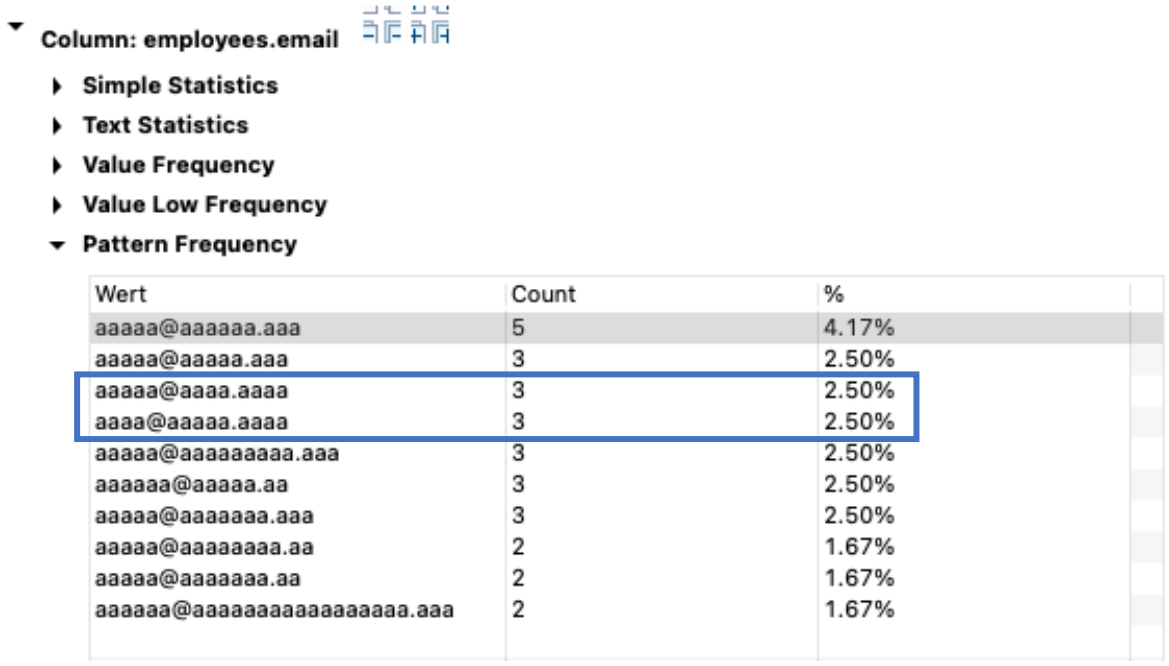



**Figure 34: Value frequency of gender in the employees table**

A quick look on the first names in the employees table reveal, that there are also women working for Super-X. Therefore, the gender column of the employees table is wrong for all female workers.



**Column: employees.email**

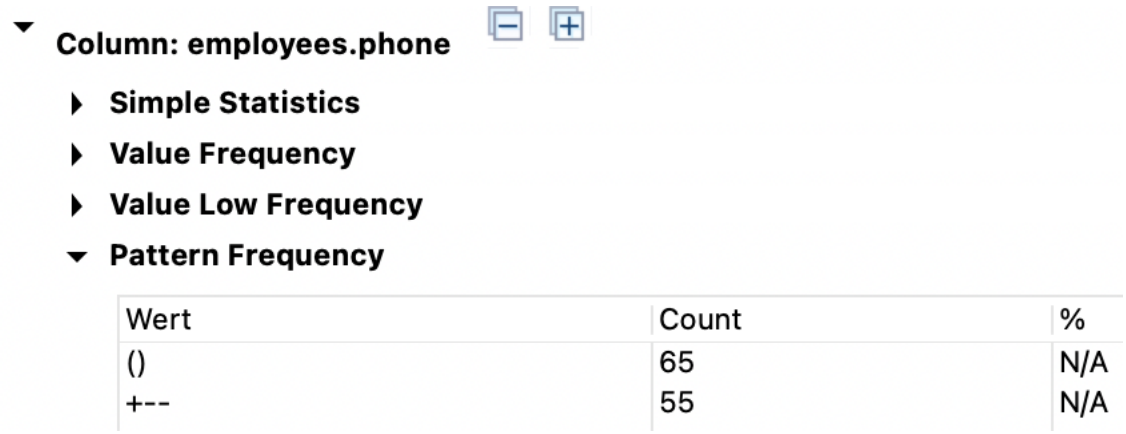
- **Simple Statistics**
- **Text Statistics**
- **Value Frequency**
- **Value Low Frequency**
- **Pattern Frequency**

| Wert | Count | % |
|---|---|---|
| aaaaa@aaaaaa.aaa | 5 | 4.17% |
| aaaaa@aaaaa.aaa | 3 | 2.50% |
| aaaaa@aaaa.aaaa | 3 | 2.50% |
| aaaa@aaaaa.aaaa | 3 | 2.50% |
| aaaaa@aaaaaaaaa.aaa | 3 | 2.50% |
| aaaaaa@aaaaa.aa | 3 | 2.50% |
| aaaaa@aaaaaaa.aaa | 3 | 2.50% |
| aaaaa@aaaaaaaa.aa | 2 | 1.67% |
| aaaaa@aaaaaaa.aa | 2 | 1.67% |
| aaaaaa@aaaaaaaaaaaaaaaaa.aaa | 2 | 1.67% |

**Figure 35: Pattern frequency of email in the employees table**

Another issue than can be seen refers to some email addresses. The pattern frequency shows two patterns with untypical endings for an email address. Since email addresses usually end with two or three characters for the top-level-domain, the marked patterns end with four. Checking the database for email addresses that end with four letters for the top-level-domain shows that there are 18 email addresses that end with ".name" and 16 email addresses that end with ".info". While ".info" is a commonly known top-level-domain, ".name" seemed to be wrong. But a short web search revealed, that ".name" actually is a valid top-level-domain (United Domains, 2021). For that reason, the column email address is okay.

A further look on the column analysis shows that there are two different formats for phone numbers in the employees table. *Figure 36* shows the pattern frequency, which reveals the two different patterns that are used in the same columns. In order to ensure a consistent format of this column, it is advisable to change one of the two formats and align them.



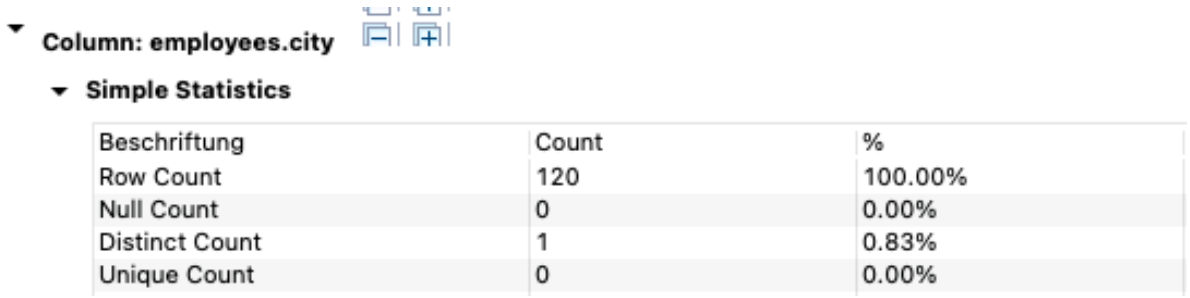
**Column: employees.phone**

- **Simple Statistics**
- **Value Frequency**
- **Value Low Frequency**
- **Pattern Frequency**

| Wert | Count | % |
|---|---|---|
| () | 65 | N/A |
| +-- | 55 | N/A |

**Figure 36: Pattern Frequency of phone in the employees table**

The next focus of this analysis will be on the address of the employees. More precisely on the city where the employees live in. The plural “cities” in the sentence before wasn’t used, since, according to the data, all employees apparently live in the same city: Berlin. This conclusion is derived from a distinct count of 1, which can be seen in *Figure 37* and more precisely a value frequency of 100% for “Berlin” (*Figure 38)*. The next paragraph will dive deeper into the analysis of the city column.



**Column: employees.city**

- **Simple Statistics**

| Beschriftung | Count | % |
|---|---|---|
| Row Count | 120 | 100.00% |
| Null Count | 0 | 0.00% |
| Distinct Count | 1 | 0.83% |
| Unique Count | 0 | 0.00% |

**Figure 37: Simple statistics of city in the employees table**

**Column: employees.city**

- **Simple Statistics**
- **Value Frequency**

| Wert | Count | % |
|---|---|---|
| Berlin | 120 | 100.00% |

**Figure 38: Value frequency of city in the employees table**

**Functional dependency analysis**

To validate if the zip codes are matching the city in the employees table, a functional dependency analysis was performed. This analysis shows a 100% match, which means that the zip code matches the city for all rows.

| Abhängigkeit | #Match | %Match | #row |
|---|---|---|---|
| zipcode-->city | 120 | 100.00% | 120 |

**Figure 39: Functional dependency of zip code and city in the employees table**

As mentioned before, the column analysis of city reveals that in all rows and for all employees, Berlin is entered as the city. Because this fact was looking suspicious, we double checked. It revealed that not all of the zip codes actually belong to Berlin. But since all rows have Berlin as the city, the functional dependency analysis was creating a pattern which linked all zip codes to Berlin and therefore, no error was detected. In this case, the functional dependency analysis gives a misleading result. To summarize the data quality problem for the city column, it can be said that all addresses, which do not belong to Berlin have a wrong city value.

# 4. Multi-dimensional design of the data mart

After defining business requirements and analyzing source data, the next step is to design the multi-dimensional model of the data mart. To come up with a suitable concept, we followed the steps of the auxiliary matrix which can be found in *Figure 40* (Ballard, Farrell, Gupta, Mazuela, & Vohnik, 2006). Therefore, the grains, dimensions and facts had to be identified. Afterwards, we used the findings to create a ME/R diagram of the conceptual data mart which was used as a base for the galaxy schema. After the schema was set up, we evaluated our model in order to find out to which extend it meets the business requirements.

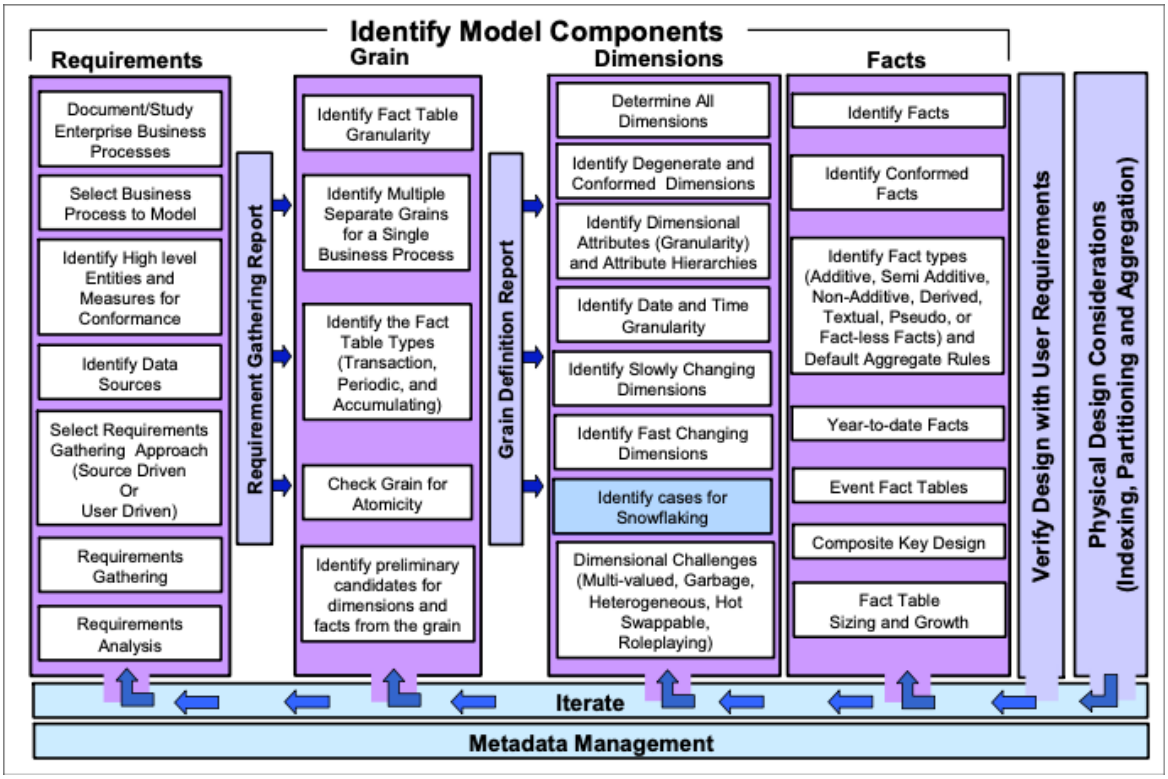



**Figure 40: Auxiliary matrix to identify model components (Ballard, Farrell, Gupta, Mazuela, & Vohnik, 2006)**

## 4.1. Identifying the grains, dimensions and facts

As a first step, we identified the granularity of the fact table according to *Figure 40*. In order to do that, we looked into the business requirements that were set and identified before. The following candidates function as the grain definitions of our fact table: order status per order of each retailer and employee, quantity per item in order, price per item in order and revenue per item in order. Since quantity per item in order and price per item in order need to be fulfilled in order to have the revenue per order in item, we decided to set two grains: 1. Order status per order of each retailer and employee as well as 2. Revenue per item in order. The level of these grains is rather high, which brings some consequences that need to be considered. From a technical perspective, a higher granularity leads to a bigger schema and therefore to a bigger data mart (Ballard, Farrell, Gupta, Mazuela, & Vohnik, 2006). This means that there could be more operational costs for tasks like the ETL process due to the size and complicatedness of the data mart. From a business point of view, 'the dimensional model should be designed at the most detailed atomic level even if the business requires less detailed data. This way the dimensional

model has potential future capability and flexibility (regardless of the initial business requirements) to answer questions at a lower level of detail.' (Ballard, Farrell, Gupta, Mazuela, & Vohnik, 2006). Like most times in life, a tradeoff between the technical and business perspective has to be made. Since answering all business requirements and being more flexible in the future is more important in our opinion, we decided for a rather high level of granularity. Besides that, we also got a very well insight in data warehouses and feel confident to overcome challenges caused by a high level of granularity.

The first idea was to use the order table as a basis for the fact table. With this approach, it would have been possible to fulfill the order status grain definition, but no other grain definition. We decided that the revenue and its connected grain definition is also important for the Sales department and enhanced this idea. The second approach we came up with, was to use the order table again as the basis for a fact table and to calculate the revenue per order within the ETL process. This could have been done by multiplying the price and the quantity within the order_items table and subsequently summing up the calculated revenues of order items per order_id. The connection to the order table then could have been made via the order_id. With this second idea it would be possible to meet the requirements for the grain definition of the order status, but not the grain definition of the revenue since we only calculated the revenue per order and not per order item. So, we came up with a third approach to use the order_items table as the basis for the fact table. Since the order_items table was listing every order item per order with its quantity and price, it was possible to calculate the revenue per order per item within the fact table and to meet the grain definition for revenue. It was also possible to add the order_status to the table via the order_id. However, since the grain definition for this table was already identified to be the one for revenue. Since 'facts that are not true to a grain definition belong to a separate fact table with its own grain definition' (Ballard, Farrell, Gupta, Mazuela, & Vohnik, 2006), we decided to create two fact tables to meet both of our grain definitions. Out of these theoretical thoughts, we created our ME/R Diagram with two cubes which can be seen in *Figure 41*.

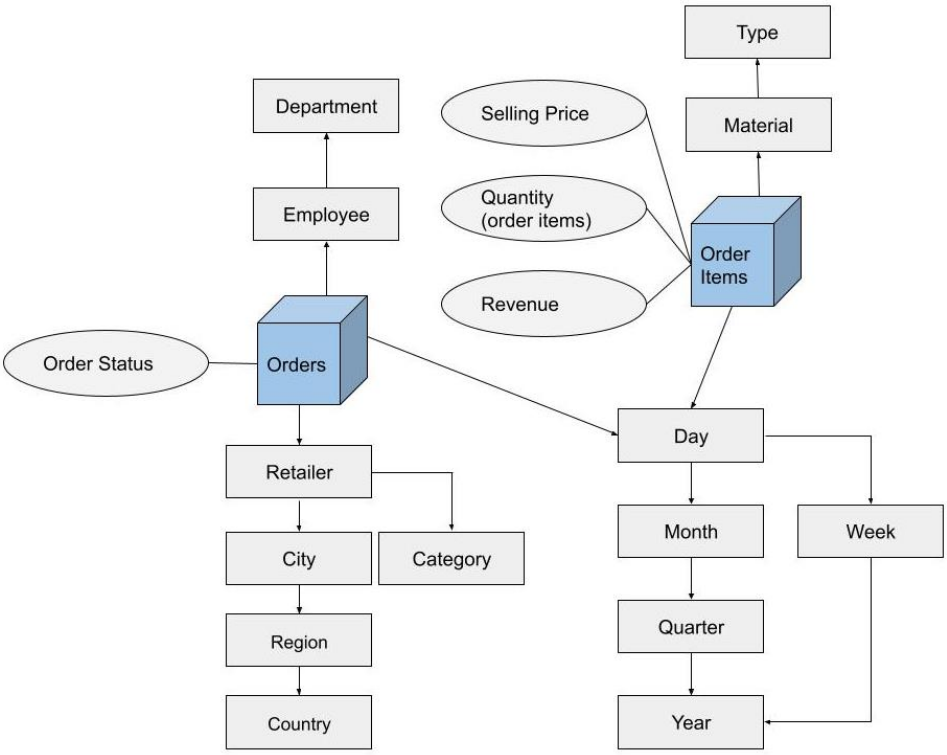



**Figure 41: ME/R model of the conceptual Sales data mart**

As seen in the ME/R diagram, we identified retailers, employees, materials and time as dimensions. All dimensions, except for time, are based on the same-named tables in the Super-X raw data. The time dimension will be created on our own. More details about this will be explained in chapter 5.1. The transformation of the ME/R diagram into a galaxy schema was done in SQL Power Architect and can be seen in *Figure 42*. It can also be seen that only one to many cardinalities are used between the dimension and fact tables. This is because of the high level of granularity in the fact tables. For example, one order item can only have one material and one timestamp. The other way around, several dates and materials are needed to be in the order_item table since this table contains several orders. As it also possible that the order item table is empty, this way of the relationship is defined as zero or more. Another observation that can be made from the galaxy schema is that we considered employees and retailers as slowly changing dimensions of type 2 and material as type 1. Therefore, we inserted the columns

version, date_from and date_to for employees and retailers, but not for the material table. -With that we can update information about employees and retailers without overwriting old transactions which avoids a misleading change of historical data. Moreover, we are able to see the history of changes. For materials, we will just overwrite old data, since they are not relevant for statistics in the Sales Data Warehouse and it is very unlikely that the type or the name of a material changes. The galaxy schema also reveals information about the additivity of the fact tables. Whereas fact_order is non-additive due to the fact that status_order is a string and cannot be summed up, fact_order_items is semi additive, because summing up price and quantity in the fact table does not lead to a meaningful result. On the other hand, summing up prices or quantities across the material dimension can lead to valuable information.

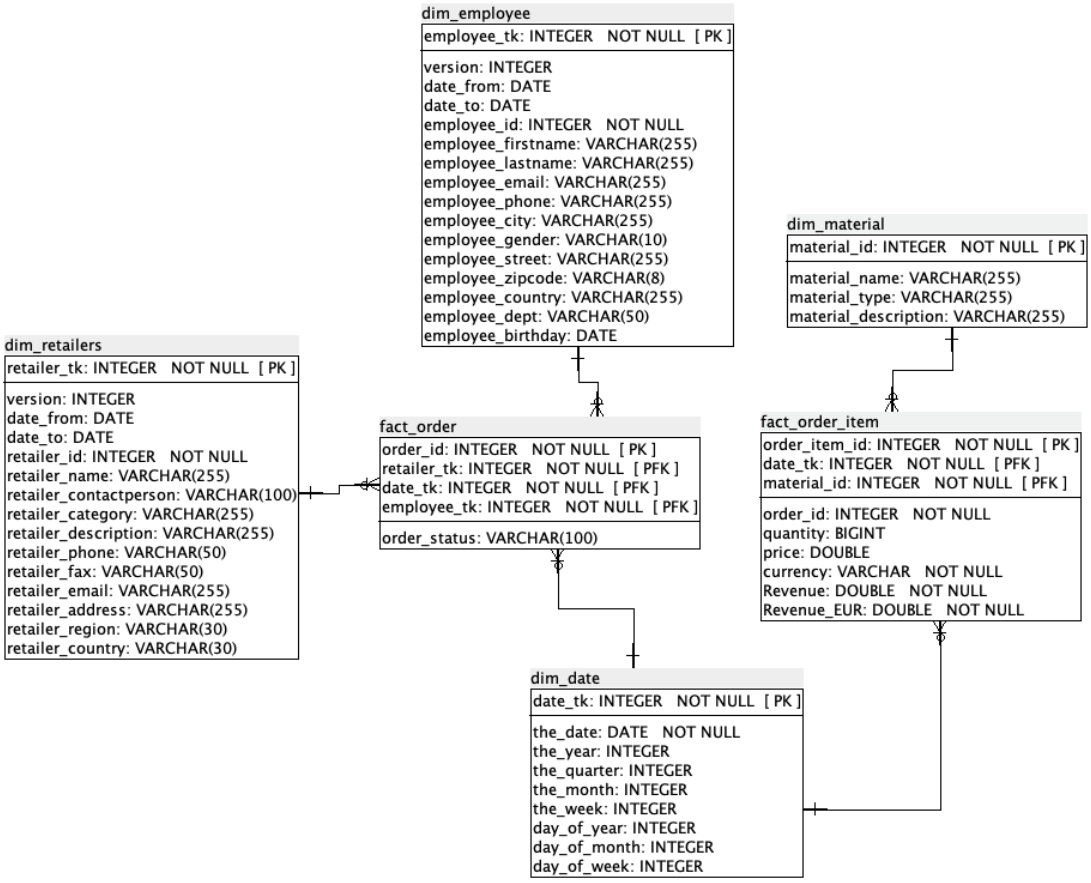



**Figure 42: Galaxy schema of the data warehouse in SQL Power Architect**

## 4.2. Verification of the model

Before implementing our model, it will be tested to see if it meets the business requirements defined in chapter 2. This testing is done to evaluate if the developed model can answer the questions that were gathered in the beginning of this project. Therefore, *Table 1* was extended by a new column to state whether the requirement can be fulfilled or not. The outcome is shown in *Table 3* below.

**Table 3: Verification of the model (business requirements)**

| No. | Business Requirement | Importance | Meets? |
|---|---|---|---|
| 1 | What is the monthly/yearly number of orders for each retailer? | High | Yes |
| 2 | What is the average employees age? | Low | Yes |
| 3 | What is the most common activity done by employees per month? How long does the activity take in average? | High | No |
| 4 | Which are the top 5 retailers that the Sales department must call the most to remind them for the order list? | Medium | No |
| 5 | Which are top ten materials according to revenue? | Medium | Yes |
| 6 | Which is the biggest category of retailers that the company is working for? How many orders per retailer were placed? | Medium | Yes |
| 7 | What are the sales for each sales employee? Who are top and bottom five employees? (excluding cancelled orders) | Medium | Yes |
| 8 | What are the sales of all materials for each month? | High | Yes |
| 9 | How much revenue was made each year? How much revenue was created with OEM products vs own products? | High | Yes |
| 10 | What is the sales volume by location for each month/year? | High | Yes |
| 11 | What is the monthly sales growth rate compared to last year? | Medium | Yes |
| 12 | What is the average quantity of material per order? | Low | Yes |

| | | | |
|---|---|---|---|
| 13 | What is the cancellation rate per month and year? Which are the top ten retailers with the highest cancellation rate? | High | Yes |
| 14 | How much is the sales loss per month? | High | Yes |

As seen in *Table 3* above, all business requirements can be met. We will start to verify the business requirements by looking at fact_order and fact_order_items (see *Figure 42: Galaxy schema of the data warehouse in SQL Power Architect*).

**Fact_order table**

Firstly, we will explain the requirements which refer to the fact_order table. Business requirement 1 asks for the number of orders for each retailer per year and month. This can be achieved by counting the order_id, grouping by retailer_tk and defining a month or year for the date_tk. In order to display more information about those retailers, a join with dim_retailers can be performed via retailer_tk as the foreign key.

Business requirement 7 questions who the employees with the most sales are. For this reason, we need to filter the fact_order table for the order_status equal to 'shipped', because only shipped orders can be seen as finished sales. Then it is necessary to group by the employee_tk and to count the order_id in order to get a list of all employee_ids with their count of completed orders. In order to see the top five, this list needs to be ordered descending. The other way around it shows the bottom five employees according to total number of shipped orders. To get the names of those employees or other information from dim_employee, a join can be done via the employee_tk as foreign key.

The next question is about the sales volume by location for each month or year, which is formulated in business requirement 10. To answer this question, fact_order needs to be joined with dim_retailers. This can be done via date_tk. Then again, we need to filter order_status for only shipped orders because canceled orders do not count as sales. Moreover, we need to apply a second filter at date_tk for the wanted year or month. Afterwards everything needs to be

grouped by whether retailer_country or retailer_region, depending on how detailed the user wants to define the location. For this purpose, even retailer_address could be used. In the end a sum needs to be applied on the order_id to actually count the number of orders for each region. To see the location with the most orders everything needs to be ordered descending.

Next up is business requirement 11, which asks to calculate the monthly sales growth rate compared to the foregoing year. This will be measured by comparing shipped orders. Therefore, we need to filter date_tk for the desired month and order_status for shipped orders. Afterwards, the sum function can be applied on order_id and the result will be the number of orders that were shipped in the desired month. This procedure can be done a second time in order to get a second number of orders from the respective month that should be the base of comparisons.

The last business requirement that can be answered with the fact_order table is business requirement number 13. This question asks to calculate the cancellation rate per month or year and per retailer. More precisely, the top 10 retailers with the highest cancellation rate. Let's start with the time approach of this questions. Cancellation rate for us in this regard will be the number of canceled orders divided by the numbers of total orders. The order_status needs to be filtered for canceled orders as a starting point. A second filter needs to be applied on date_tk to specify the desired time for which the cancellation rate will be calculated. Then, the count function needs to be applied to order_status to have the total count of cancelled orders in the desired time frame. To derive the total number of orders to perform the calculation for the cancellation rate, the same method as described before can be used but the filter for order_status needs to be deleted. This will return the total number of orders for the given date_tk. To find out about the cancellation rate for specific retailers, just one small adjustment needs to be made to the before mentioned methodology, which is adding the retailer_tk as a filter. Then again, a filter for 'Canceled' needs to be applied to order_status. To get the total number of orders, this filter just needs to be removed.

**Fact_order_item table**

The first question that can be answered with fact_order_item is business requirement 5, which asks for the top 10 materials according to revenue. For that, material_id and revenue_eur are important. The fact_order_item table needs to be grouped by the material_id to only show unique IDs. After that, the sum function needs to be applied to revenue_eur to show the revenue for each material_id. In the very end the table needs to be ordered by the sum of revenue_eur in a descending order. If an additional information about the materials such as the name is wanted, a join with dim_material could be made via material_id.

Corresponding to business requirement 8, we will focus on materials again. This time, we will check the quantity of each material sold per month. In order to achieve this, the first step is to filter date_tk by the respective month that should be shown. Then we need to group by the material_id and to apply the sum function over the quantity to show the sum of the quantity for each material_id in the respective month that was filtered before. Again, if an additional information from dim_material is desired, a join via material_id is possible.

Business requirement 9 is also addressing the materials, which includes the revenues created by the sales of OEM products in comparison to the ones of the own products for each year. Whereas joining fact_order_item and dim_materials was optional before, it is necessary now to derive the two wanted numbers. After joining via material_id, two filters need to be applied. Firstly, the date_tk for the year has to be filtered and secondly, a filter for the material type has to be created. Afterwards, we need to apply the sum function over revenue_eur in order to see whether the revenue was created by selling OEM products or non-OEM Products. These numbers can be compared to fulfill the desired outcome. To answer the other question from business requirement 9, we only have to set a filter on date_tk for the desired year and then apply the sum function on revenue_eur.

The average quantity per order is asked in business requirement 12. This question can be answered by just applying the average function to quantity in the fact_order_item table. In order to get the desired result, a filter for the wished material_id needs to be set.

To find out how many orders per retailer were placed, as it is asked in business requirement 6, the count function needs to be applied to order_id. After grouping by retailer_tk, the desired outcome will be shown. If another information about the retailers is needed, a join with dim_retailers can be done via retailer_tk. The second question of business requirement 6 can be answered by only looking at the dimension tables. To find the biggest category of retailers, we only need to group by retailer_category in dim_retailers to get unique values and then get the count of retailer_category. To show the biggest category first, we need to sort the outcome in a descending order.

For business requirement 2 the average age of all employees in the sales department is the number we are searching for. Here, we only need dim_employee to answer it and filter for the department of Sales. Afterwards, it is necessary to subtract the date of the query from the employee_birthday and to calculate the average to get the desired number.

A very special and the most complicated business requirement is number 14: ‘How much is the sales loss per month?’. The reason for this is that we need to combine both fact tables in order to answer it. We are aware that this should be avoided, but we did it for the following reasons. On the one hand, this business requirement was very important to us from a Sales perspective and therefore, we did not want to delete it. On the other hand, to avoid joining two fact tables, we would’ve needed to put everything in one fact table. But this would have led to not being able to answer a lot of other important business requirements. So, we decided to make an exception for this one business requirement. As already mentioned, the first step to get the desired outcome, is to join the two fact tables. This can be done via the order_id. Second step is to filter by order_status and only include canceled orders. Then the sum function needs to be applied to revenue_eur. This gives the total loss in revenue due to canceled orders. This outcome could be

made more meaningful by adding more filters to see for example the sales loss at a specific year or month, or to see the sales loss for different materials or retailers.

The business requirements number 3 and 4 need to be emphasized, since these questions cannot be answered by the dimensional model. Instead, the information can be derived from the event log of the Sales department. Therefore, process mining is needed to gather the needed information. This will be done later in our report in *chapter 6.*

**Table 4: Verification of the model (maintaining history)**

| **History Requirement** | **Action** |
|---|---|
| Employee changes department | Add a new row |
| Retailer changes address | Add a new row |
| Material changes type | Overwrite |

Next to the verification for business requirements, also the requirements for handling history need to be validated (Ballard, Farrell, Gupta, Mazuela, & Vohnik, 2006). An overview can be seen in *Table 4* above. Since it is common to change core data for employees and retailers sometimes, we identified those as slowly changing dimensions type 2. Therefore, we added the columns date_from, date_to as well as version to dim_employee and dim_retailers. So, whenever there is a change in the data source, our model will create new rows for new versions. This is important for certain business requirements to be not falsified because of changes. For example, if a retailer changes its region and this change would just be overwritten in the respective row for the retailer in dim_retailers, the count of all previous orders that were made in the old region, would now be mapped to the new region. This would distort our statistics about orders by location. Another table we identified to be possibly affected by a change of historical data is dim_material. We defined this table as slowly changing dimension type 1. This means that in case of changes in the source data, the respective row will be overwritten. Since changes are not expected and would not affect any of our business requirements, overwriting will not be a problem.

After verifying the model, we found out that all business requirements can be met and therefore, no adjustments need to be made. This means that our multidimensional model can be implemented.

# 5. Implementation

In this chapter, we cover all the steps for data mart implementation including the creation and loading of the data. In *Figure 42* it is shown how our galaxy schema looks like. After running a forward engineering SQL script in Power Architect, the schema was created in the team1 database. From DBeaver, we got the schema shown in *Figure 43*. It can be seen that technical keys are set as primary keys for the employee and retailer dimension. The reason for that is the beforementioned second type of slowly changing dimension in these tables. For the material dimension, we only set the material_id as a primary key since we have the type one of a slowly changing dimension here. For both fact tables, the primary key can also be seen in bold, as a combination of different technical keys (from dimensions) with a specific identifying key for each table.

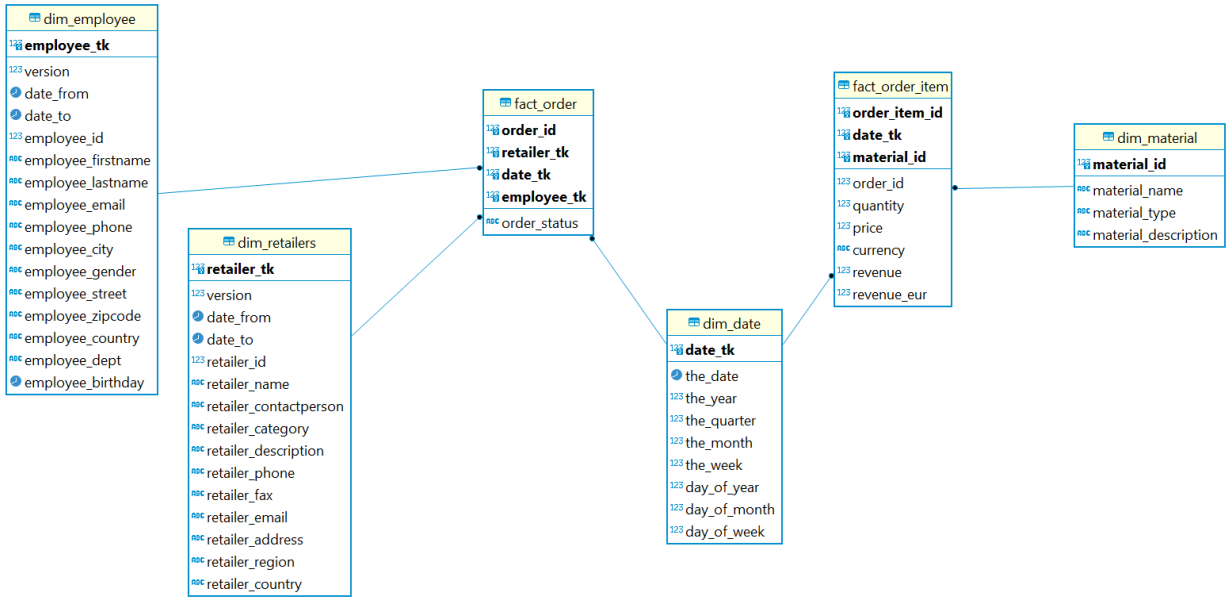



**Figure 43: Galaxy Schema in DBeaver**

## 5.1. ETL process

After the galaxy schema was loaded to the database, the population of the tables by using ETL (Extraction, Transformation, Loading) processes had to be done as a next step. It covers the process of how the data are loaded from the source system to the data warehouse. Extraction is the first step of the ETL process where data from different sources like txt, XML, Excel files or other sources are collected. The extraction of the data is followed by the data transformation as the second step in data integrations. This includes the compiling, converting, reformatting and cleaning in the staging area to load the data into the target database as the final step of the ETL process.

### 5.1.1 Transformation

In this subchapter, all undertaken transformations are presented which were part of the ETL process for both dimensions and fact tables. To do so, Pentaho was used as a tool.

**Date ETL**

This is the date on which the row is loaded in the data warehouse. It has nothing to do with the business itself. It simply documents the exact date. Once the date stamp is added, it is never updated for the life of the database. It defines the first time the row arrived in the database. It is important to mention, that during a batch load the date of the beginning of the process should be stored in the variable and the exact same date should be used for one instance of the batch process. Upfront, we created rows for each day in a time interval. Each row includes date_tk, the_date, the_year, the_quarter, the_month, the_week, day_of_year, day_of_month, day_of_week fields which will be explained in detail later in this chapter.

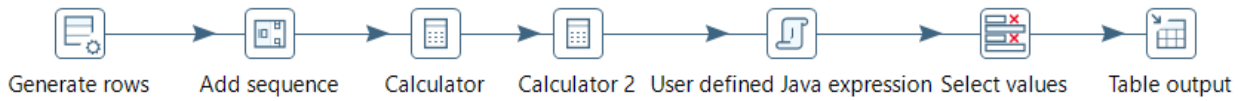
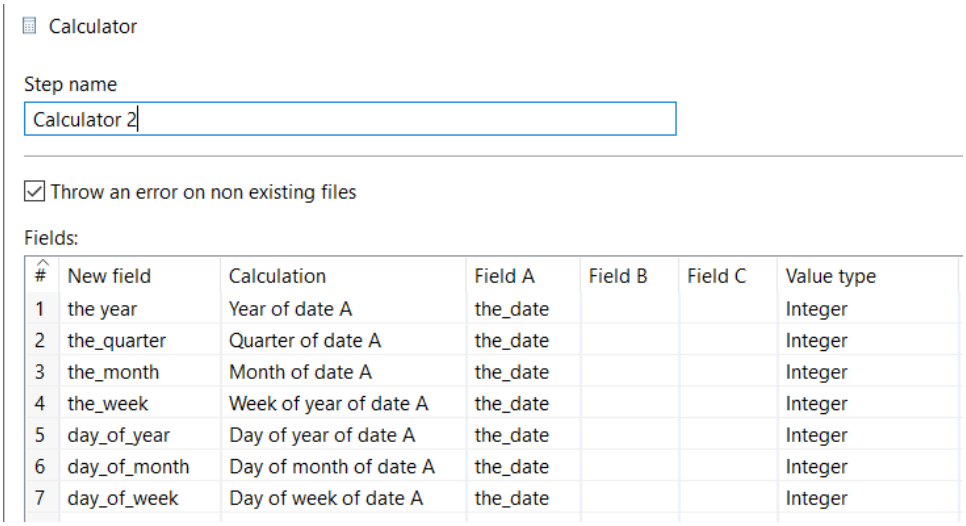



**Figure 44: Dim_date ETL structure**

Starting with the generation of rows with the date format yyyy-MM-dd as the output and continuing with adding a sequence of numbers from 0 with increment 1. Using the calculator function, we added both newly created elements and saved it as the_date. Using the calculator again, we created some new columns with different calculation functions as listed below:


Calculator

Step name

Calculator 2

Throw an error on non existing files

Fields:

| # | New field | Calculation | Field A | Field B | Field C | Value type |
|---|---|---|---|---|---|---|
| 1 | the year | Year of date A | the_date | | | Integer |
| 2 | the_quarter | Quarter of date A | the_date | | | Integer |
| 3 | the_month | Month of date A | the_date | | | Integer |
| 4 | the_week | Week of year of date A | the_date | | | Integer |
| 5 | day_of_year | Day of year of date A | the_date | | | Integer |
| 6 | day_of_month | Day of month of date A | the_date | | | Integer |
| 7 | day_of_week | Day of week of date A | the_date | | | Integer |

**Figure 45: Calculator step of dim_date**

Afterwards, we used a Java expression to create date_tk as an integer with the format yyyyMMdd. This field was used as a relation key later. The last transforming step for dim_date was the selection of the values which selects and loads only the relevant fields to our dim_date table. Below is the result of the first three rows after the select values step was executed.

| # | date_tk | the_date | the year | the_quarter | the_month | the_week | day_of_year | day_of_month | day_of_week |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 20100101 | 2010-01-01 | 2010 | 1 | 1 | 1 | 1 | 1 | 6 |
| 2 | 20100102 | 2010-01-02 | 2010 | 1 | 1 | 1 | 2 | 2 | 7 |
| 3 | 20100103 | 2010-01-03 | 2010 | 1 | 1 | 2 | 3 | 3 | 1 |

**Figure 46: Result of select values of dim_date ETL**

The last process of the dim_date ETL is the table output to load the transformed data into our dim_date table. We assumed that this ETL process will only run once a year to fill 365 new rows, so this ETL process will not run-in daily transaction job. Therefore, we didn't create a change data

capturing job for this ETL process. Instead, it would be possible to run it manually and change the generated row values as the start date which we want to add, as shown in *Figure 47*.

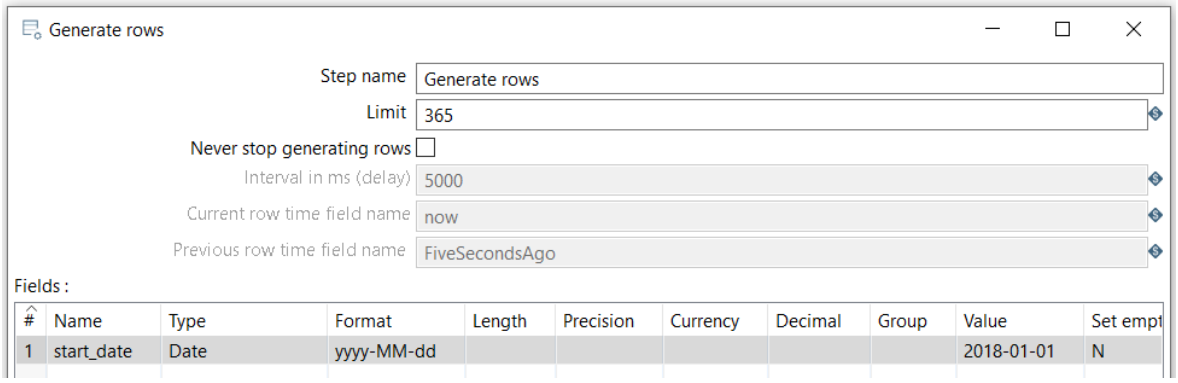



**Figure 47: Generate rows**

**Employee ETL**

This is the ETL process for extracting employee data from the superx_development employee table. A 'database lookup' step was used to get the department_name for each employee from the superx_development department table. Afterwards, we used the 'dimension lookup/update' step to compare the inserted data with existing data in the dim_employee table. If the data do not already exist, a new row is inserted with a new employee_tk. Otherwise, a new row with a new employee_tk, a version value with the latest version +1 and the current date as the date_from is created. At the same time the date_to value of the old entry is changed to the date of the previous day.

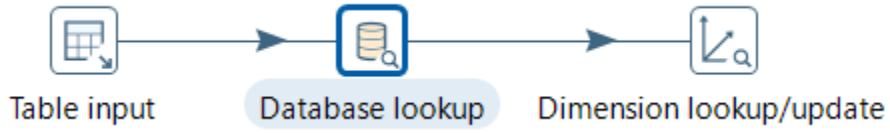



**Figure 48: Tr_dim_employee ETL structure**

This tr_dim_employee ETL process can be used to do the first migration of data as well as for daily transactions.

**Retailer ETL**

This is the ETL process for extracting retailer data from the superx_development retailer table. Firstly, we created a cnt variable using a modified java script to split the address (using split fields with comma delimiter). This step counts how many commas (,) appear in the address fields. Then, in the switch/case step, the cnt result will determine the next step. If the cnt is equal to 3, the process will continue to the first stream. If it is equal to 4, then the process will continue to the second stream.

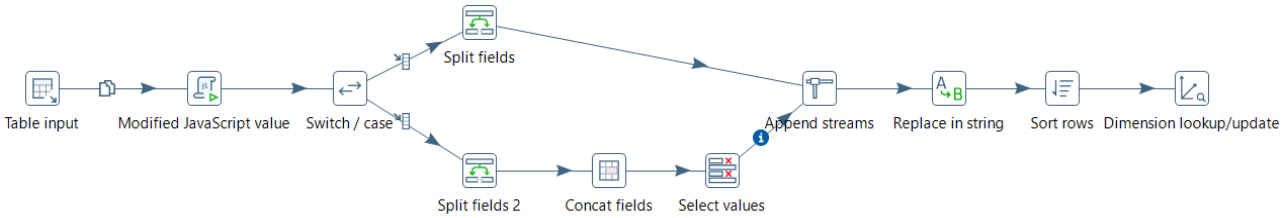



**Figure 49: tr_dim_retailer ETL structure**

The first stream starts with a 'split fields' step which separates the text between the comma in the address field into four different columns: retailer_address, retailer_zip, retailer_region, and retailer_country. The second stream is similar to the first one. The only difference is that the step of separating the text between the comma in the address field splits the text into five different columns: retailer_address, retailer_address2, retailer_zip, retailer_region, and retailer_country. Afterwards, we joined the retailer_address with the retailer_address2 with a 'concat fields' step and assigned it to retailer_address. This stream ended with a 'select values' step which eliminates other unused fields such as retailer_address2. The 'append streams' step is used to merge both streams back into one stream. In this step, we have to make sure that the field we are going to merge has the same amount and sequence. The next step is 'replace in string' which is used to replace a designated string into our intended string. These are:

1. Retailer category e-shop into online
2. Retailer category offline retailer into offline
3. Retailer phone (999) 999-999 into null

4. Retailer fax number (999) 999-999 into null
5. Retailer country U.S.A. into USA. Since there are two different formats, we chose to use 'USA'.
6. Retailer country Deutschland into Germany. Since both means the same country.

Then we sort the id of the retailer in ascending order and load the data to the dim_retailer table in our database using 'dimension lookup/update'. This ETL process can be used to do the first migration of data as well as a daily transaction ETL.

**Material ETL**

This ETL process is used to extract material data from the superx_development material table. It directly inserts new data or updates existing data in the dim_material table in our data mart. If there are any future changes, our existing data will be directly updated. So, this ETL process can be used for the first migration of data as well as for daily transaction ETL.


**Figure 50: Tr_dim_material ETL**

**Extra order ETL**

Integrating extra orders in the database required some transformation which had to be done beforehand. The following steps were necessary:

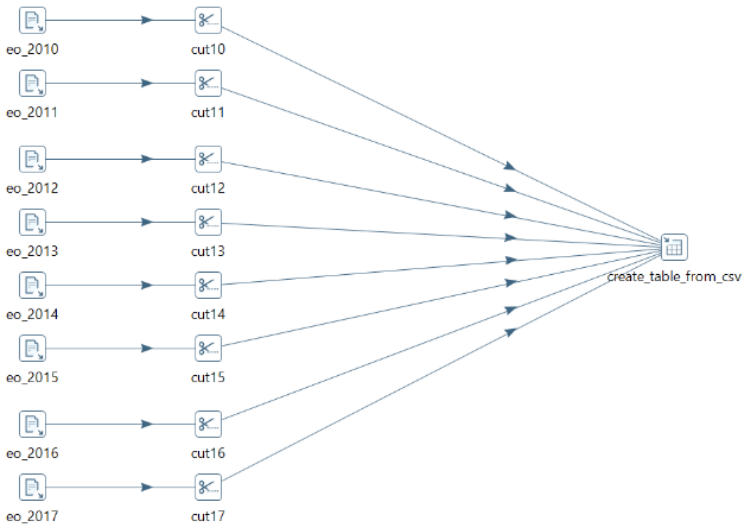



**Figure 51: Transformation to merge all csv files in one table**

Firstly, all different extra orders are merged in one table. During this process we realized that the date type is string and it also contains a time zone. Therefore, we used ‘cut step’ to remove the time zone part.

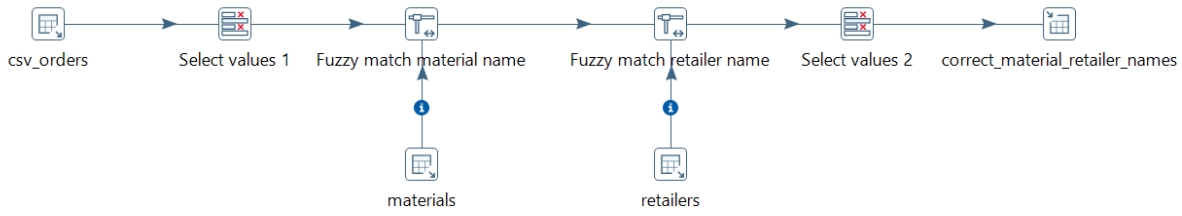



**Figure 52: Transformation to correct misspelled names**

Secondly, a transformation to correct the names which are misspelled was created. The 'Fuzzy match' step runs an algorithm known as Jaro Winkler, which returns the name to which the misspelled one is most similar. In this way, all the data are cleaned. In the step 'select values 1' we also change the data type of our fields from string (all of them are strings) to the corresponding type that we will need in the final fact table.

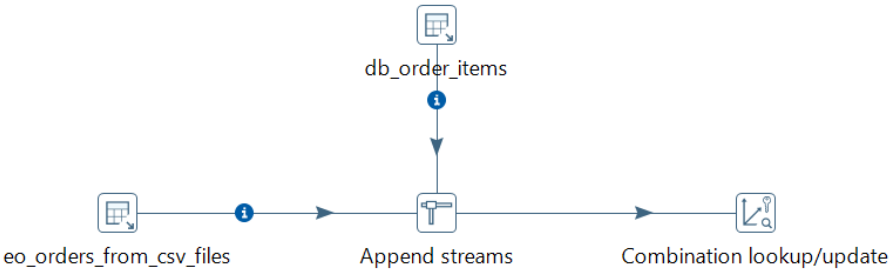



**Figure 53: Transformation to merge all order items together**

Last but not the least, the extra order items from the csv files are merged with the order items from the database using 'Append Streams'. Additionally, some other things are adjusted in this step. For each order item in the csv files, an order_id is assigned. To identify the right order_id, the timestamp and the retailer_id are used as comparing elements. This means that to each extra order item the same order id, that exists for other order items done in the same month/year and from the same retailer, is assigned. The code from eo_orders_from_csv_files looks like this:

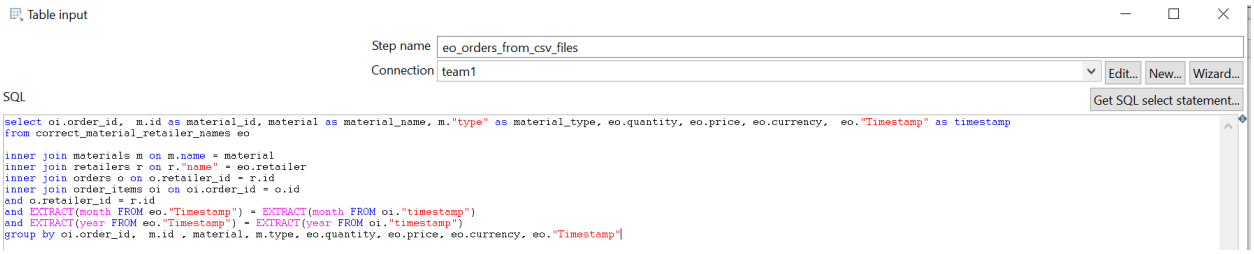



**Figure 54: Identifying order_id for extra orders**

In the previous analysis from Talend, it was shown that there are some null values within the currency column in the orders table. To solve this, a left join between order_items, orders,

materials and retailers is performed. Since the retailers table contains the currency for each retailer, a new currency column is produced, which contains no null values.

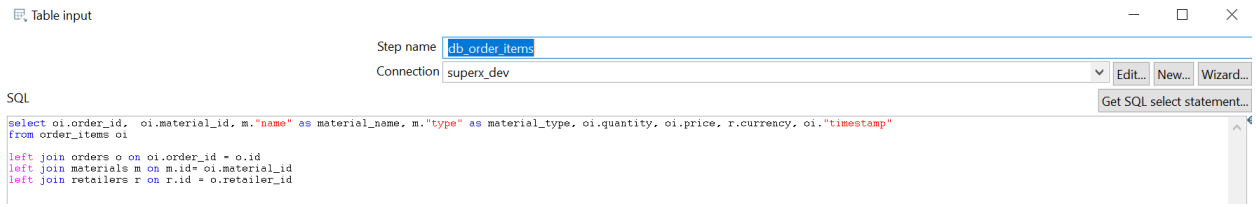



**Figure 55: Completing null currency values with the right one**

In the end, a new order_item_id is created for every order item. The result is a table called all_order_items which will be used for creating the fact table fact_order_item.

### Fact order items ETL

This is the ETL for extracting the order_item data from the superx_development database and the extra_order_items data from the legacy system which was previously joined into our all_order_items table (see *Figure 56*). The first step is 'table input' which extracts data from the all_order_items table. Afterwards, we add new fields: date_tk(which transforms the timestamp column into the format yyyyMMdd), date_currency (which transforms the timestamp column into the format yyyy-MM-dd), to_currency and decimal. 'Set field value to a constant' is used to set the to_currency column values to EUR and rounds the value to three numbers after the comma.


**Figure 56: Fact_order_items ETL structure**

The fifth step is 'web services lookup'. Here we use a free API from http://currencyconverter.kowabunga.net/converter.asmx?WSDL with the operation of 'GetConversionRate' and set the inputs using: the existing currency field as 'CurrencyFrom',

to_currency as ‘CurrencyTo’, date_currency as ‘RateDate’. Afterwards, we assign the output conversion to ‘GetConversionRateResult’.

Web Service in GetConversionRateResult

| # | Name | WS Name | WS Type |
|---|---|---|---|
| 1 | currency | CurrencyFrom | string |
| 2 | to_currency | CurrencyTo | string |
| 3 | date_currency | RateDate | dateTime |

**Figure 57: Web services lookup input setting**

Web Service in GetConversionRateResult

| # | Name | WS Name | WS Type |
|---|---|---|---|
| 1 | GetConversionRateResult | GetConversionRateResult | decimal |

**Figure 58: Web services lookup output setting**

The ‘revenue_eur’ step is the calculator action which consist out of two sequential events. The first event is calculating the revenue (quantity* price) with ‘GetConverionRateResult’ and puts the calculation result to ‘revenueEUR’. The second event is rounding ‘revenueEUR’ with the variable ‘decimal’ which was created before. The last transformation selects only the needed fields and loads them to the fact_order_items table in our data mart.

**Fact order ETL**

For the fact order table, a simple ETL process for loading the data is created. The steps include: selecting the right columns from the orders table, looking up the dim_employee and dim_retailer table to get the corresponding technical key, creating the technical key for dim_date, selecting only the columns we need and finally loading all the data to the fact_order table.

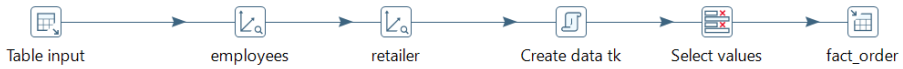



**Figure 59: Fact_order ETL structure**

### 5.1.2 Creating and running jobs

After creating all the transformation, it is time to create jobs. In this way, we were able to automate the process of running each transformation one by one. Firstly, a job for running all transformations for the dimension tables was created as it can be seen in *Figure 60.*

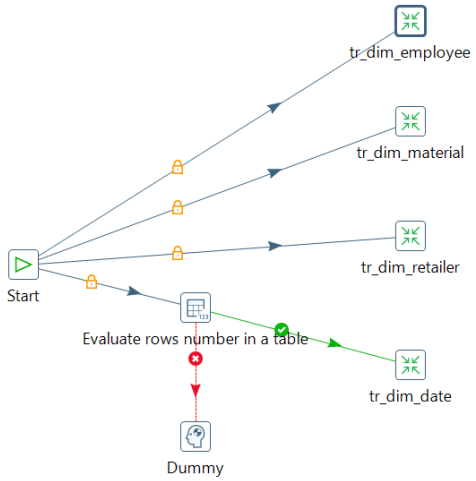



**Figure 60: Job load dimensions in parallel**

Secondly, all transformations for creating all_order_items are merged in one job, as shown in *Figure 61*.

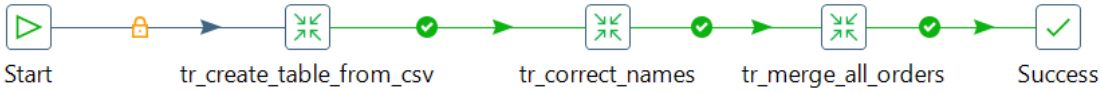



**Figure 61: Job creates all order items**

Then, transformations for both fact tables are run in parallel.

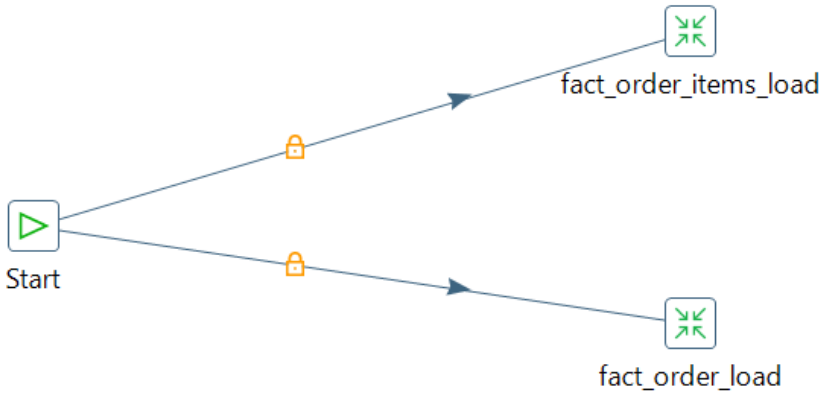



**Figure 62: Job for running job transformations on parallel**

In the end, all these steps are merged in one final job, as shown in *Figure 63*. After running the job, the data were loaded in the team1 data mart and were ready to be used for the next steps.

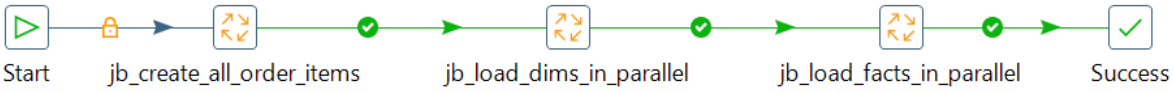



**Figure 63: Final job for loading data**

Moreover, this chapter addresses the change data capturing of the fact tables. In our case, we use the queryable change data method which means that the changed data come directly from the data source via a query. For every new element to be added in the fact table it is checked if the id already exists in the database. If yes, it doesn't have to be inserted again. Therefore, a transformation called variable_fact_orders was created as a first step. It returns the maximum value of the primary key from the fact table.

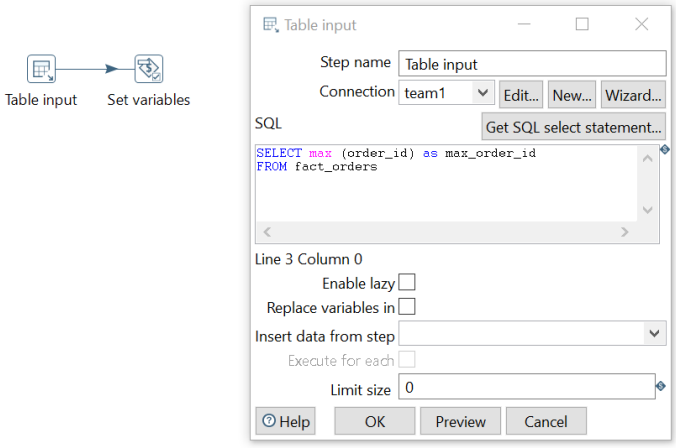



**Figure 64: Variable Creation**

Then, a SQL script was written to ensure that there is no element in the fact table which has an order_id greater than the maximum.

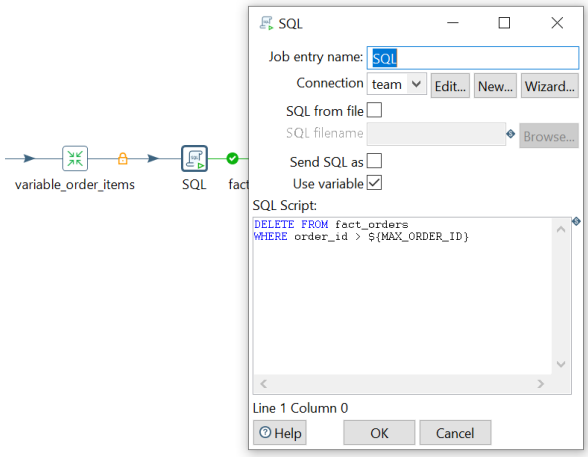



**Figure 65: SQL Script**

Afterwards, the transformation that we used to load the data in the fact_orders is transformed as it is shown in *Figure 66*.

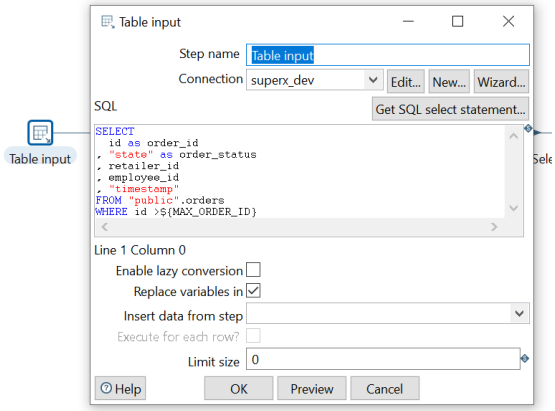



**Figure 66: Transformation for fact table using cdc**

Finally, the job looks like in *Figure 67*.

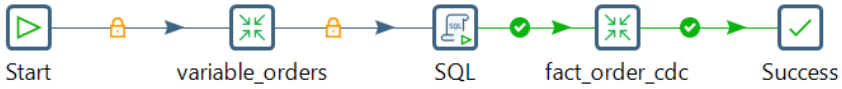



**Figure 67: Job for change data capturing (Fact Order Table)**

By following exactly the same steps, for the fact_order_items table, we created the job shown in *Figure 68*.

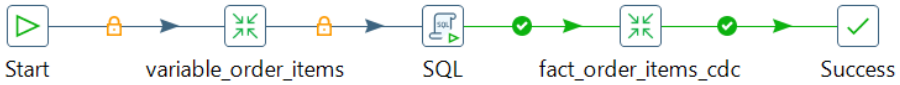



**Figure 68: Job for change data capturing (Fact Order Items Table)**

In the end, the final job, is transformed as below.


**Figure 69: Job with cdc fact tables**

## 5.2. KPIs visualization

After creating the Sales data mart and loading the transformed data to it, we were able to start with the data analysis by visualizing the KPI's to answer the most important questions which are interesting for Sales (see chapter 2). For the visualization, Tableau and Power BI were used as tools. Both are compared in *Chapter 5.3*. To avoid redundant work, but still be able to compare both tools, we decided to visualize the KPI's we can extract from the fact table of order items with Tableau and the KPI's regarding the fact order table with Power BI. An exception is the material dashboard which was created with both tools to enable a better comparison. In some cases, we also made cross references between both fact tables to make the dashboards more informative. In this chapter, different screenshots were taken from the dashboards to explain and underline our findings regarding most questions which were mentioned within the business requirements.

**Tableau**

*Figure 104* of the appendix section A shows the dashboard which was created with Tableau and mainly includes data from the order items fact table. The dashboard can be divided in two sections. Firstly, there is a section about the revenue of Super-X which is one of the most interesting KPI's for Sales. Two charts were created for displaying the revenue per year and per month to show trends over a period of time and also fluctuations within one year. Secondly, the dashboard contains a section about the sold quantities (sales) per year and per product. Since the data only reach until the beginning of 2017, the charts of the yearly KPI's end with the year 2016 to avoid misunderstandings. Therefore, the data from 2017 about the total revenue and the quantities sold are displayed separately within the dashboard. For all figures, only the shipped orders were taken into consideration since the canceled orders do not lead to a higher revenue or sales and that is why including them would lead to a misleading result.

The first chart of the dashboard (see *Figure 70*) displays the total revenue of Super-X per year. The green part contains the shipped orders which means that this part provides information

about the real revenue made. The red part contains the canceled orders which can be understood as lost revenue for Super-X due to cancelations.

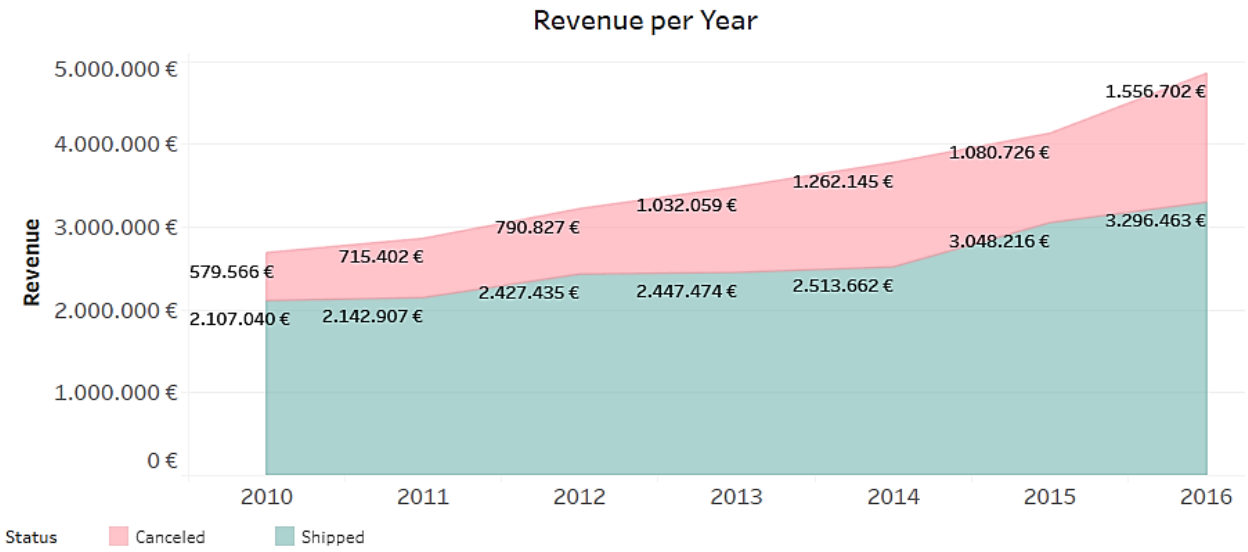



**Figure 70: Total revenue per year**

In 2016 about 3.3 Million Euro of revenue were made. This is a growth rate of more than 8 % compared to the previous year. By filtering for the different product types, we found out, that Super-X makes more revenue by the sale of their own products. From the mentioned 3.3 Million Euro of revenue in 2016, about 90% (3 Million Euro) were made with the toy cars and only 10% (0.3 Million Euro) with OEM products. Considering the whole period from 2010 to 2016, an average growth rate of 7.99% was calculated. The highest growth rate regarding the revenue was accomplished in 2015 (21.27%). Looking at the graph, it seems that the rate of cancellations is increasing as well. Referring to the 'lost' revenue, it seems that nearly half as much orders are canceled as orders are shipped. This means that about one third of all orders are canceled. But since it is better to look at the absolute number of orders and their status instead of the revenue, this part will be explained in more detail later with the help of the Power BI dashboards.

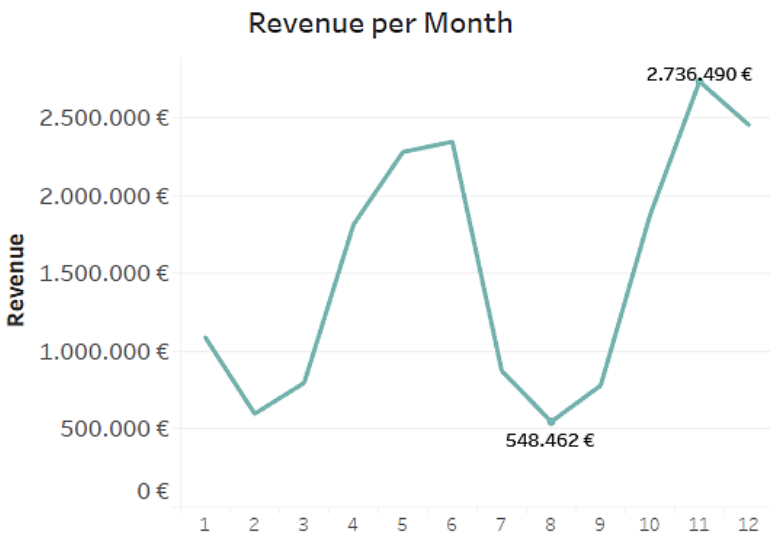



**Figure 71: Total revenue per month**

*Figure 71* shows the revenue per month (including the revenue of all years). It gives insights into the fluctuations within one year. The graph has nearly the shape of a sinus curve with two peaks and two low points. This indicates that there are low and high seasons for Super-X. The most revenue is made in the end of the year around November. It can be assumed that the products of Super-X are a popular Christmas gift which leads to this peak in revenue. Another peak is visible for late spring between May and June. The least revenue is made in February and August. The whole summer from July to September is a low season for Super-X which is probably caused by the absence of people who are on vacation. After having a deeper look into it, there was no difference found between OEM products and own products.

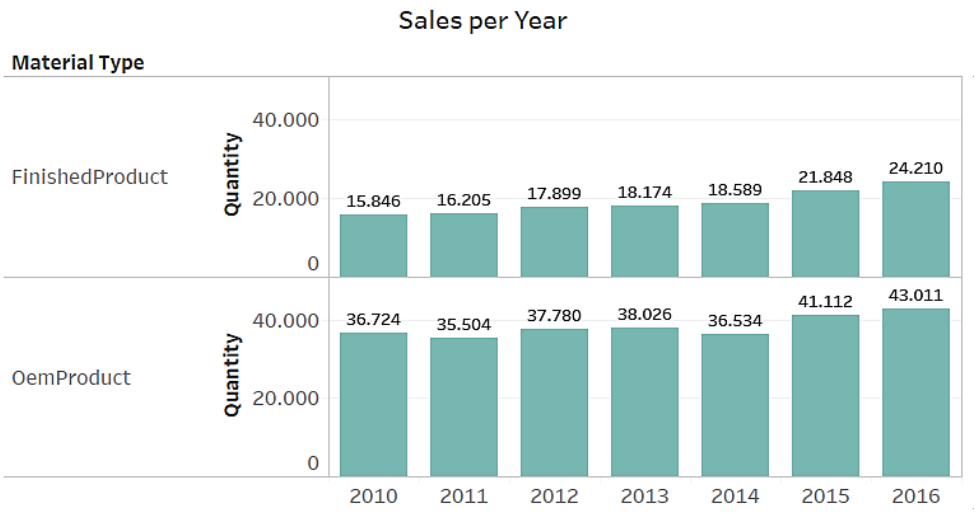



**Figure 72: Annual sales per product type**

After analyzing the revenue, we want to have a look into the quantities sold per product type and per product. In *Figure 72* it can be seen that Super-X sells much more quantities of OEM products compared to their own manufactured products (finished products). This can be easily explained since the OEM products are accessories like batteries, controllers or stickers which are much cheaper than a toy car. Furthermore, the graph reveals that there is an increase in sales for both product types but with a higher growth rate for the own manufactured products. Considering the whole period from 2010 to 2016 it is visible that the sales of the finished products increased every year while the sales of OEM products had some fluctuations. In comparison to the previous year, Super-X sold 10.81% more toy cars (finished products) in 2016 which leads to the increase of the revenue as shown before.

Sales per Product

| Material Name | 2010 | 2011 | 2012 | 2013 | 2014 | 2015 | 2016 | 2017 |
|---|---|---|---|---|---|---|---|---|
| Super-X Booster Beast | 4.088 | 4.474 | 5.530 | 6.036 | 6.579 | 8.319 | 9.704 | 668 |
| Motor 12V | 3.567 | 3.907 | 4.480 | 4.804 | 4.822 | 5.719 | 6.514 | 475 |
| Super-X Buggy Champ | 3.773 | 4.207 | 4.741 | 4.998 | 5.536 | 6.361 | 7.326 | 456 |
| Booster Beast Logo Stickers | 3.569 | 3.711 | 4.121 | 4.391 | 4.386 | 5.302 | 5.626 | 387 |
| Remote Controller 2-Channel 2MHz | 3.462 | 3.260 | 3.609 | 3.667 | 3.710 | 4.084 | 4.455 | 335 |
| Receiver 2-Channel 2MHz | 3.070 | 3.098 | 3.478 | 3.514 | 3.455 | 3.955 | 4.141 | 324 |
| Receiver Channel 1MHz | 3.250 | 3.248 | 3.454 | 3.490 | 3.530 | 4.011 | 4.283 | 320 |
| Super-X Monster Truck | 2.998 | 3.066 | 3.365 | 3.329 | 3.175 | 3.662 | 3.847 | 290 |
| Ni-Cd Battery 12V 300mAh | 3.007 | 2.968 | 3.082 | 3.052 | 2.902 | 3.210 | 3.322 | 245 |
| Remote Controller 1MHz | 2.818 | 2.793 | 2.953 | 2.895 | 2.788 | 3.071 | 3.086 | 218 |
| Tire 20 mm | 2.944 | 2.762 | 2.973 | 3.168 | 2.912 | 3.324 | 3.308 | 212 |
| BIPM Experts Logo Stickers | 2.818 | 2.575 | 2.595 | 2.454 | 2.266 | 2.439 | 2.441 | 183 |
| Monster Truck Logo Stickers | 2.932 | 2.585 | 2.628 | 2.357 | 2.139 | 2.246 | 2.205 | 177 |
| Super-X BIPM Expert Racer | 2.862 | 2.608 | 2.550 | 2.394 | 2.087 | 2.210 | 2.183 | 173 |
| Offroad Logo Stickers | 2.714 | 2.350 | 2.343 | 2.295 | 1.982 | 2.028 | 2.012 | 149 |
| Buggy Logo Stickers | 2.573 | 2.247 | 2.064 | 1.939 | 1.642 | 1.723 | 1.618 | 142 |
| Super-X Offroad Car | 2.125 | 1.850 | 1.713 | 1.417 | 1.212 | 1.296 | 1.150 | 61 |

**Figure 73: Annual sales per product**

Figure 73 provides information about the popularity of the products sold by Super-X. By far, the Super-X Booster Beast is sold the most with 9.704 pieces in 2016. Also, the Super-X Buggy Champ reaches a high number in sales. The table shows that both products were among the most popular every year with a continuous increase in sales and therefore, they are kind of a cash cow for Super-X. The most sold OEM product is the Motor 12V with 6.514 units in 2016. The last position with the lowest number of sales is the Super-X Off-road Car with only 1.150 pieces in 2016. The sales of this product decreased every year and therefore, Super-X could consider dropping it. But for making a clear decision on that, additional KPI's like the profit per sold unit must be considered. Also, the Super-X BIPM Expert Racer has a continuous decrease in sales.

**Power BI**

In power BI, we used a dynamic data presentation to answer more business requirements with a simple dashboard. Before we created a dashboard, we needed to create additional field information for the employee dimension with a power query:

1. Employee name: In the power query editor of dim_employee dimension, we chose to add and custom a column. As can be seen in *Figure 74* below, we concatenated employee_firstname with employee_lastname and separated both with a space sign.

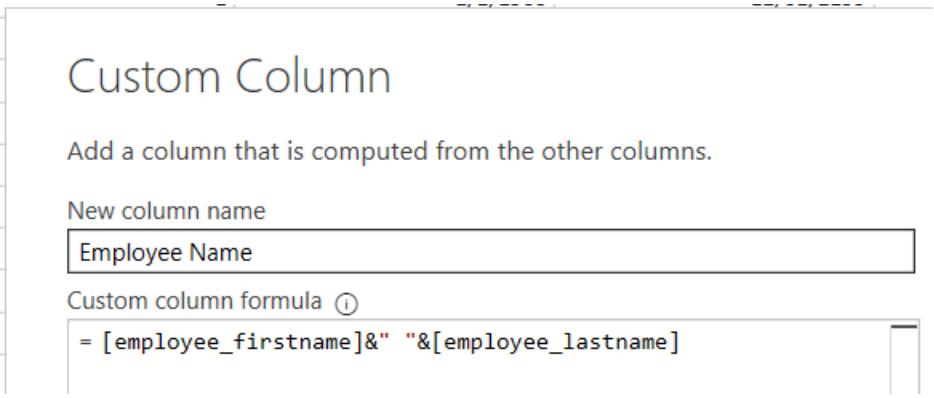



**Figure 74: Custom column for employee name**

2. Employee age: In the power query editor of dim_employee dimension we did the following, as shown in *Figure 75* and *Figure 76*:
    1. Select the column that contains DOB (employee_birthday)
    2. Go to Add Column Tab,
    3. Under “From Date & Time” section, select Date
    4. Then select Age.
    5. Select the new Age column,
    6. Under “From Date & Time” section, select Duration
    7. Select Total Hours, to create a new field of total years.

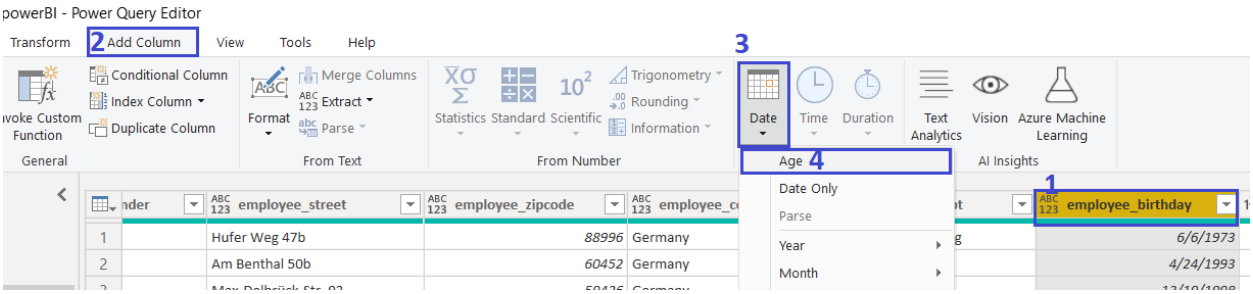



**Figure 75: Steps to create age column**

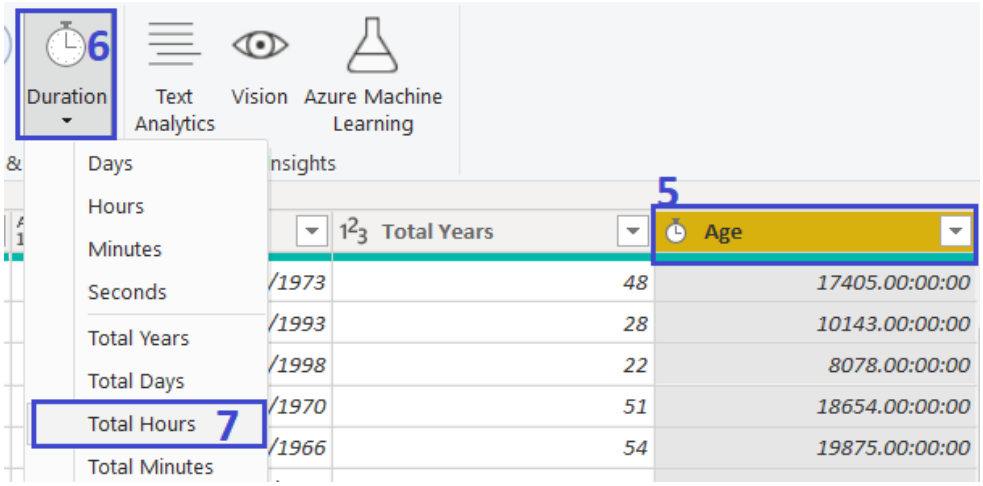



**Figure 76: Steps to create employee_age column**

We also created an additional measurement using the DAX functions:

1. Count of order_id for Canceled = CALCULATE(COUNTA('fact_order'[order_id]),'fact_order'[order_status] IN {"Canceled"})
2. **Cancelation rate** = if(([Count of order_id for Canceled]) / count(fact_order[order_id]) = 1 , 0 , ([Count of order_id for Canceled]) / count(fact_order[order_id]))
3. Total order LY = calculate(count(fact_order[order_id]), SAMEPERIODLASTYEAR(dim_date[the_date].[Date]))
4. **Order Growth** = if((count(fact_order[order_id])-[Total order LY])/count(fact_order[order_id]) = 1 , 0 , (count(fact_order[order_id])-[Total order LY])/count(fact_order[order_id]))

In our dashboard, only cancelation rate and order growth rate will be shown.

Lastly, we used a cross table reference between the order_status of the fact_order table and the fact_order_item table. In power query, we merged the queries for fact_order_item with a left outer join with fact_order, using order_id as the matching column. The steps are shown in *Figure 77*. Afterwards, we chose to add only order_status to our fact_order_item table.

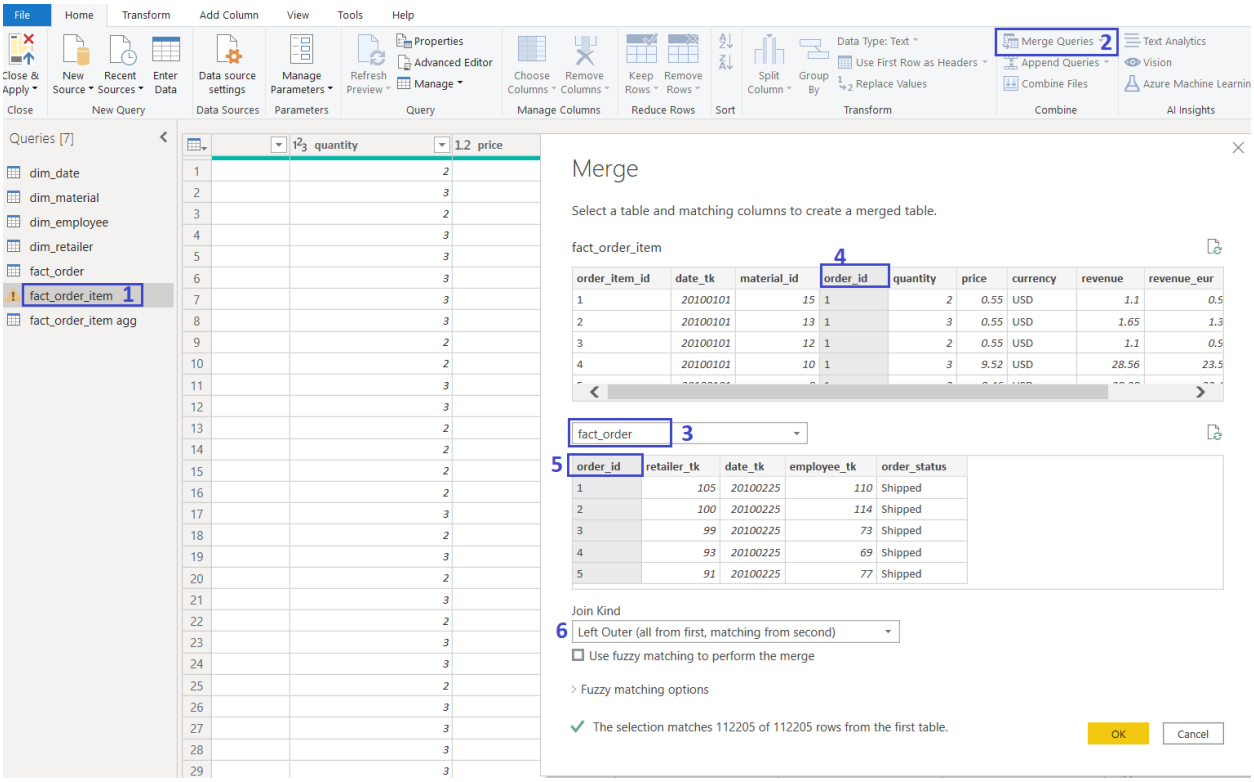



**Figure 77: Steps merge queries**

We separated our Power BI dashboard into 3 different reports: Retail order dashboard, Employee order dashboard and Material revenue dashboard.

**Retail order dashboard**

In the retail order dashboard, we mainly focus on the number of orders placed by retailers. *Figure 78* shows the first view of the retailer order dashboard without any selections or filters.

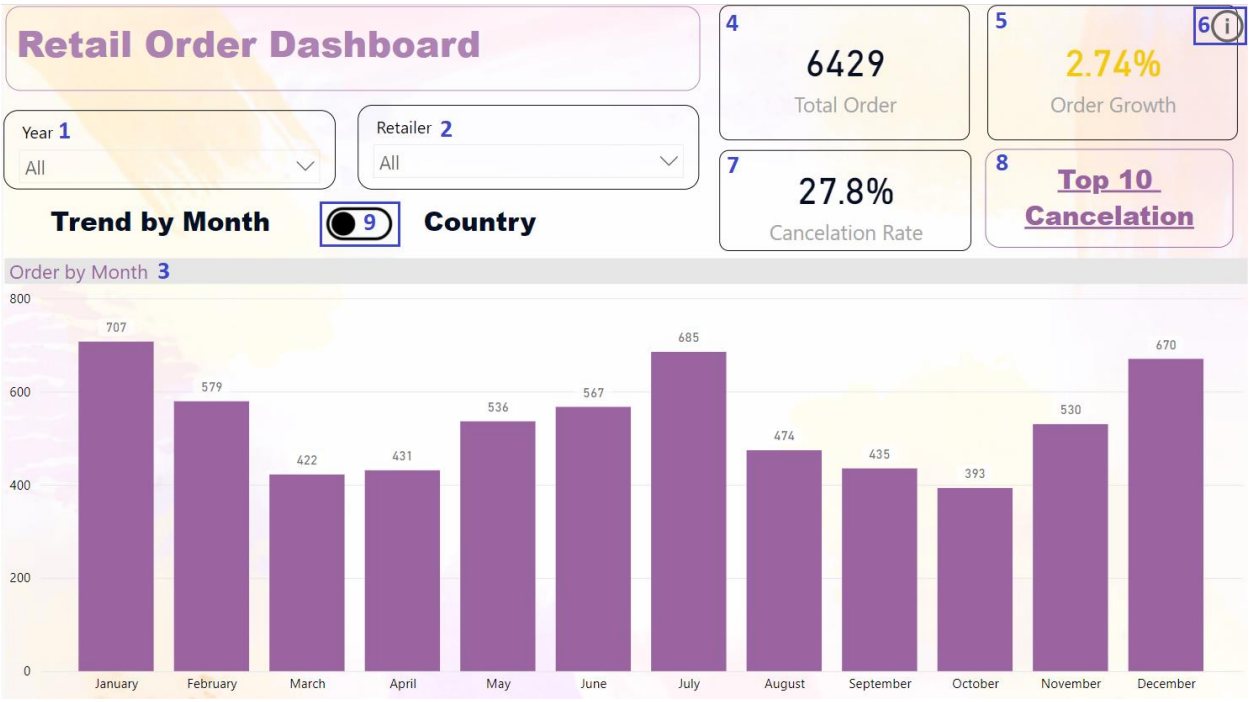



**Figure 78: Retail order dashboard**

We used several filters and elements to create a switch function manually since no switch function has been developed in Power BI, yet. Below is the list of them:

1. Year filters: This filter is used to filter the year across this retail order dashboard. When selected, all other elements will be changed according to the selected year. This year's filter is set to be able to filter multiple years, as shown in *Figure 79*.

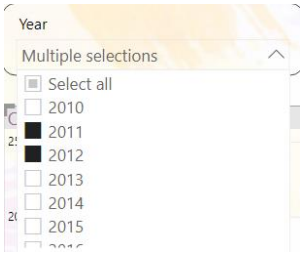



**Figure 79: Year filter multiple selection**

2. Retailer filter: This filter also has ability to filter whole visualizations in the retail order dashboard and has the same ability to filter multiple selections of retailers.
3. Order by month trend bar chart: This bar chart shows the total orders received and shipped by month. By this chart we can answer the business requirement to show the number of orders for each retailer per month and per year. An example is showed in *Figure 80*. For the year 2012 the total number of shipped orders is 1.279. Additionally, the number of shipped orders for each month can be seen in the trend bar chart.

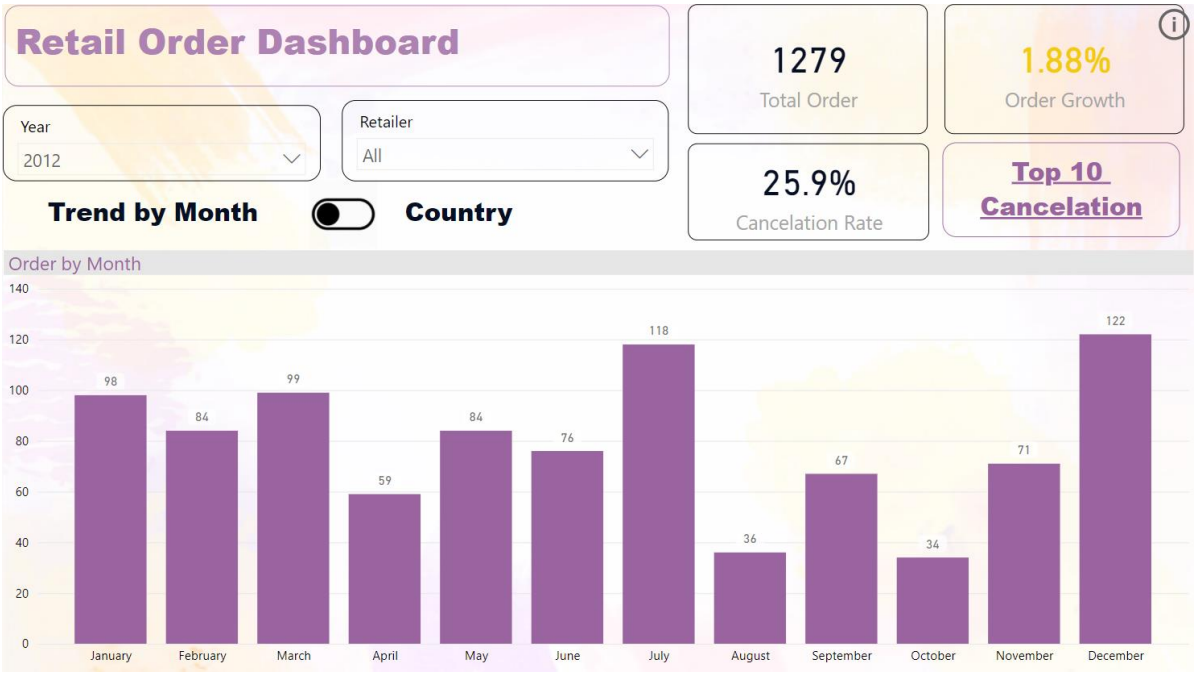



**Figure 80: Total order per month 2012**

4. Total order: It shows the number of orders placed by the retailers. This element can be filtered by year and retailer as well as month.
5. YTD order growth: This element shows the order growth per year, which means this visualization will not be affected by the selection of the month. This visualization contains extra information by setting the threshold for a color:
   - Below 0% = Red
   - 0% - 5% = Yellow
   - >5% = Green

6. Information icon: This icon shows the retail category and the number of retailers.

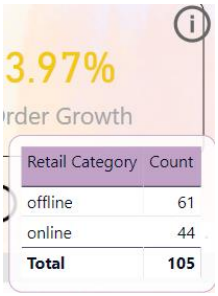



**Figure 81: Retailer category table from information icon**

We can answer the business requirement of which retailer category has the bigger contribution. In *Figure 82,* when filtering for the offline retailer category, we can see the total number of orders is 745 for 2012. For the online retailer category 534 orders were placed. Additionally, we get to know that the orders placed by offline retailers increased by 2.15% compared to the previous year.

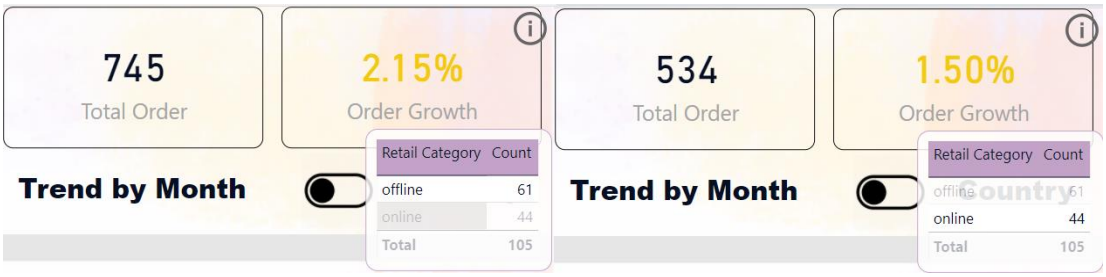
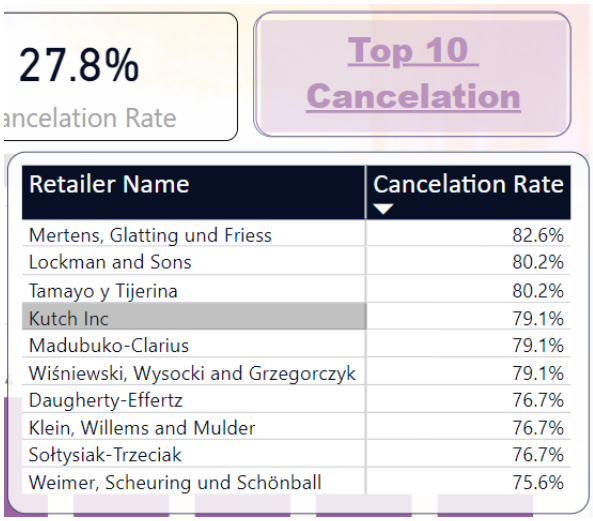



**Figure 82: Comparison of retail category**

7. Cancelation rate: Shows the percentage of the cancelations made by each retailer per month, if we filter for the retailer and the month.
8. Top 10 cancelation rate button: By clicking the Top 10 Cancelation button, it will show the list of the top 10 retailers with the highest cancelation rate. This table can be affected by the year filter but not by the retailer filter.




| Retailer Name | Cancelation Rate |
|---|---|
| Mertens, Glatting und Friess | 82.6% |
| Lockman and Sons | 80.2% |
| Tamayo y Tijerina | 80.2% |
| Kutch Inc | 79.1% |
| Madubuko-Clarius | 79.1% |
| Wiśniewski, Wysocki and Grzegorczyk | 79.1% |
| Daugherty-Effertz | 76.7% |
| Klein, Willems and Mulder | 76.7% |
| Sołtysiak-Trzeciak | 76.7% |
| Weimer, Scheuring und Schönball | 75.6% |

**Figure 83: Top 10 cancelation rate by retailer table**

9. Switch from trend by month to country: By clicking the button we get a view of the orders per country. With that we can compare which country had the highest number of orders on a specific day. For example, on 02.2012 the most orders were made in the USA which can be derived by the size of the circle.

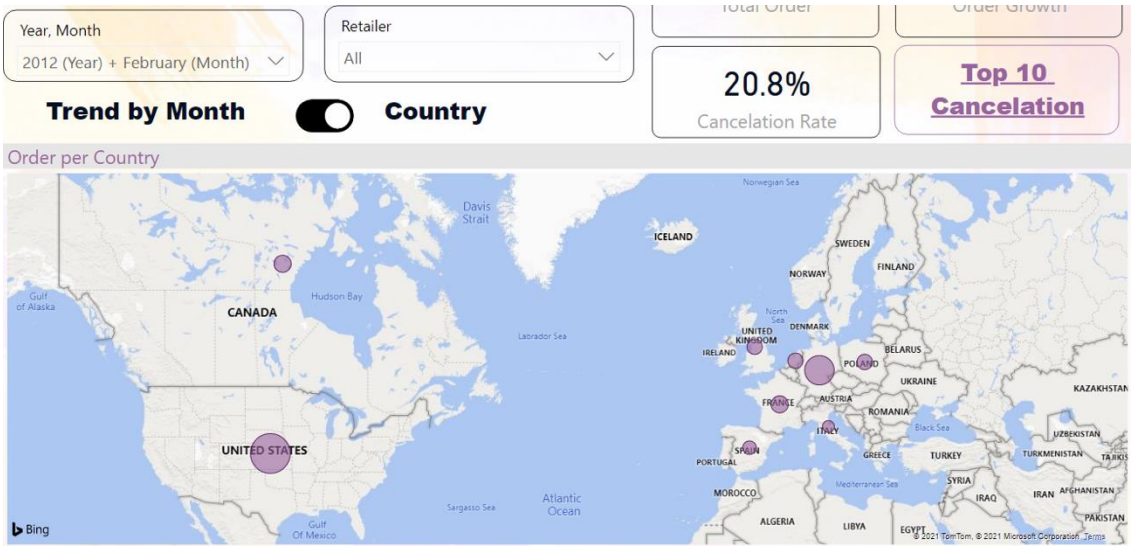



**Figure 84: Retailer orders by country with map view for 02.2012**

After analyzing the retailer order dashboard, we can conclude that from 2010 to the beginning of 2017, the total cancelation rate was up to 27.8%. With Mertens, Glatting und Friess as the retailer with the highest order cancelation rate with about 82.6% over the whole period of time. Comparing the two retailer categories, more products are sold to offline retailers than to online retailers. Finally, most orders were placed in the United States followed by Germany and France.

**Employee order dashboard**

In the employee order dashboard, we mainly focus on the number of orders created by the Sales employees. *Figure 85* shows the first view of the retailer order dashboard without any selection or filter.

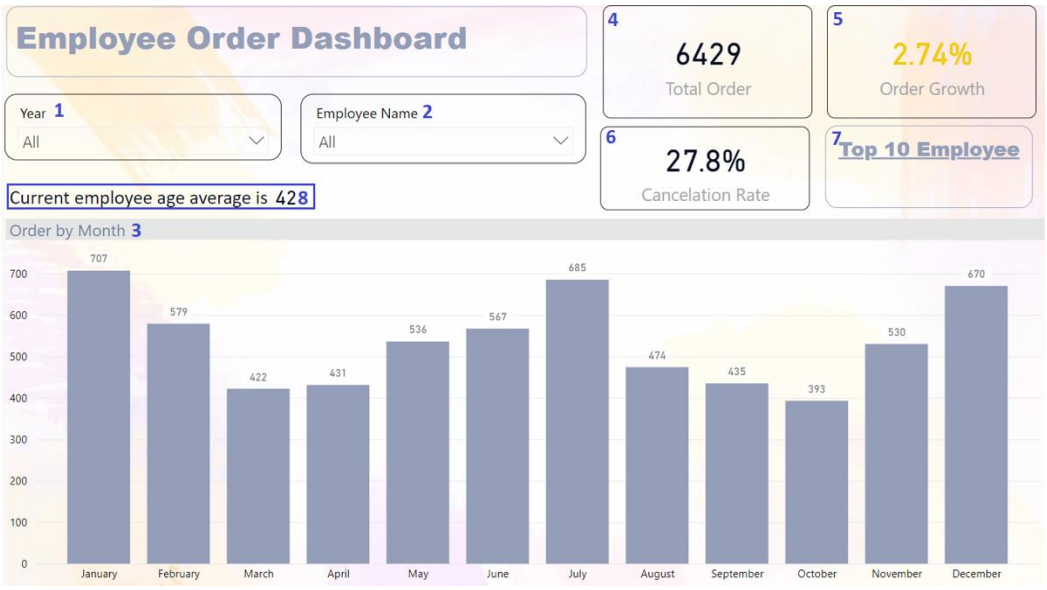



**Figure 85: Employee Order Dashboard**

In this dashboard we use a similar template like for the retailer order dashboard. We also use several filters, elements and a top 10 button. Below is the list of them:

1. Year filters: This filter is used to filter year in this whole employee order dashboard. When it is selected, all other elements will be changed according to the selected year. This year filter has been set to be able to filter multiple years as well.

2. Employee filter: This filter also has the ability to filter whole visualizations in the employee order dashboard and has the same ability to filter multiple selections of employees. The selection list of this employee filter has been filtered to show only sales employees, which are only 28.

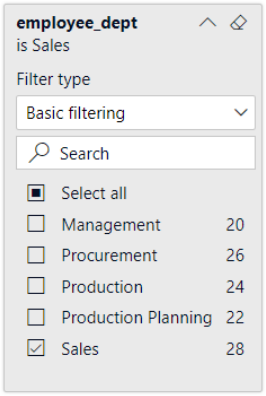



**Figure 86: Sales department employee filter**

3. Order by month trend bar chart: This bar chart shows the total number of monthly orders received and shipped. By this chart we can answer the business requirement regarding the number of orders created per employee, per month and per year. An example can be found in *Figure 87*. For year 2012, the total orders shipped by Charlotta Brettschneider are 29 orders which is 17.24% (34 orders by her were shipped in 2011) less than in the previous year.

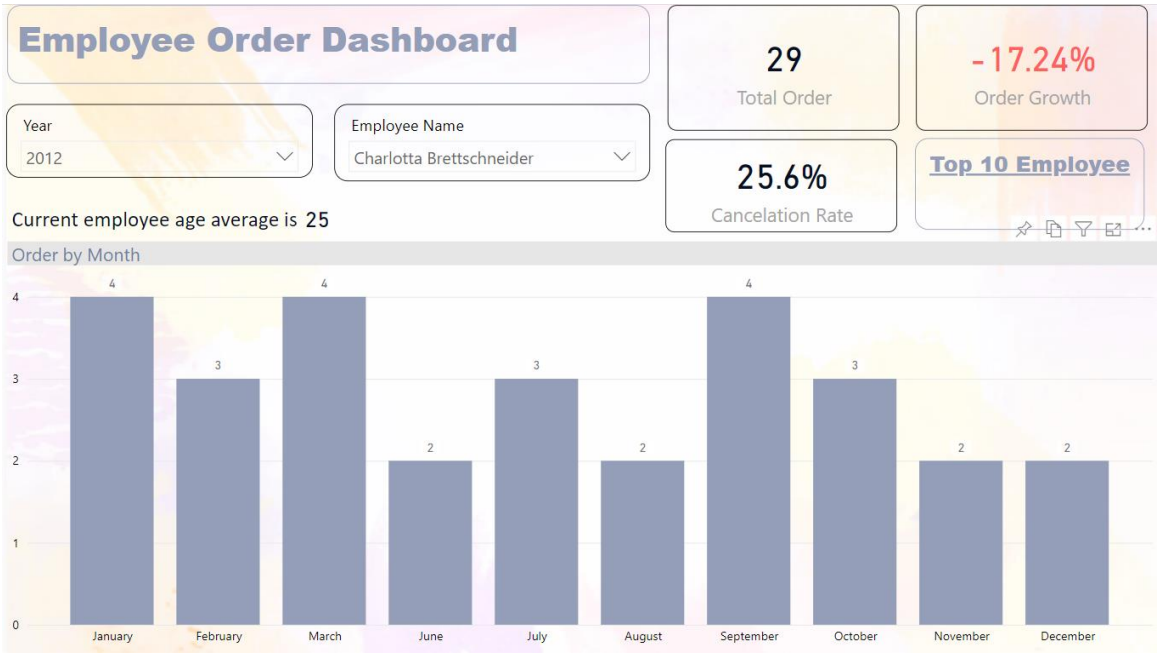
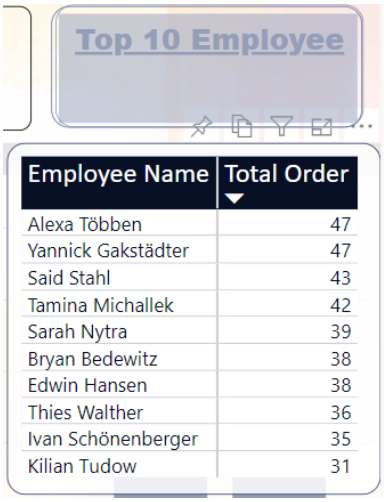



**Figure 87: Total order per month 2012**

4. Total order: It shows the number of orders created per employee. This element can be filtered per year and employee as well as month.
5. YTD order growth: This element shows the order growth rate per year, which means this visualization is not affected by the selection of the month from trend bar chart. This visualization provides additional information by setting the threshold for a color. We have set it as:
   - Below 0% = Red
   - 0% - 5% = Yellow
   - >5% = Green
6. Cancelation rate: Shows the percentage of cancelations of created orders per employee per month. This can be done by filtering for the employee and month.
7. Top 10 employee button: By clicking the top 10 employee button, it will show the list of the top 10 employees with the highest order creation.




| Employee Name | Total Order |
|---|---|
| Alexa Többen | 47 |
| Yannick Gakstädter | 47 |
| Said Stahl | 43 |
| Tamina Michallek | 42 |
| Sarah Nytra | 39 |
| Bryan Bedewitz | 38 |
| Edwin Hansen | 38 |
| Thies Walther | 36 |
| Ivan Schönenberger | 35 |
| Kilian Tudow | 31 |

**Figure 88: Top 10 Employee table 2012**

8. Employee average age: This text shows the average age of the selected employees. This element can be affected by the year filter and employee filter. By selecting one employee, this element shows the employee's current age. For example, from *Figure 87* we can see Charlotta Brettschneider is 25 years old today.

After analyzing the employee order dashboard, we can conclude that that Said Stahl is the best performing Sales employee with a total number of 274 orders received throughout the years. Additionally, we can say that the average age of all Sales employees is about 42 years.

**Material revenue dashboard**

In the material revenue dashboard, we mainly focus on the revenue sales in EUR made per material/product. *Figure 89* shows the first overview of the material revenue dashboard without any selections or filters.

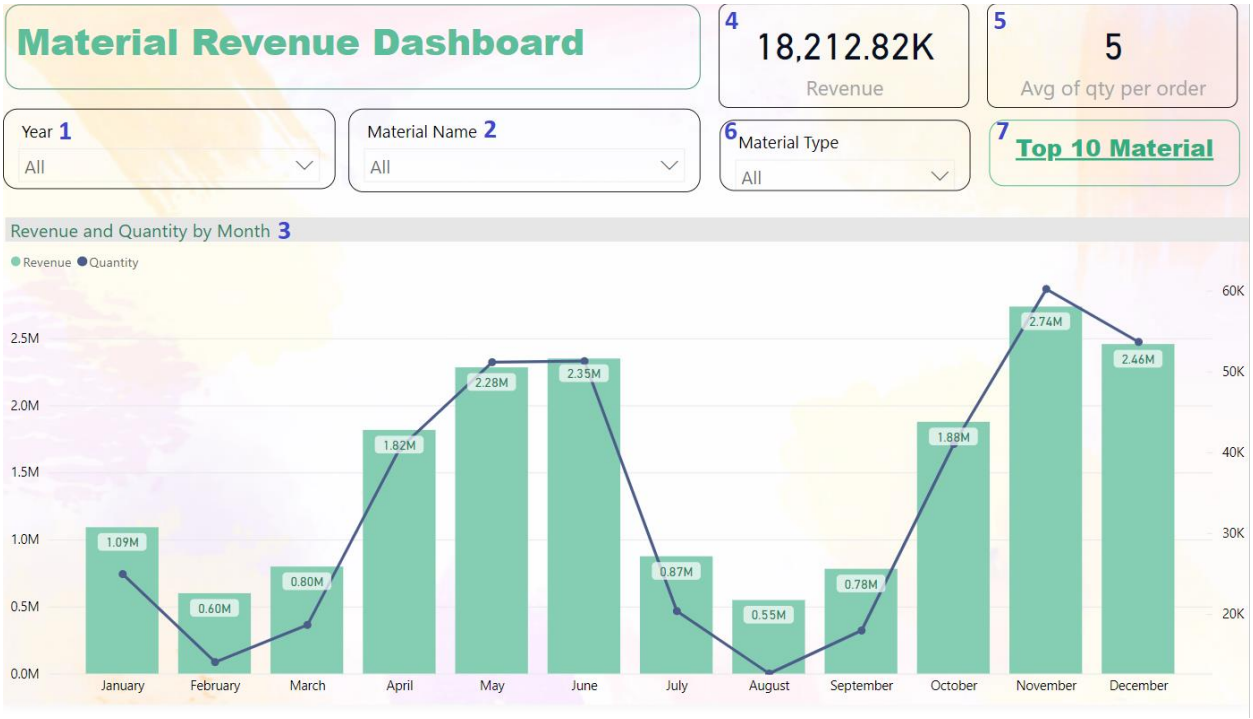



**Figure 89: Material revenue dashboard**

In this dashboard we use the similar template like for the other dashboards. We also use several filters, elements and the top 10 material button. Below is the list of them:

1. Year filters: This filter is used to filter for years across this material revenue dashboard. When selected, all other elements will be changed according to the selected year. This year's filter has been set to be able to filter multiple years as well.
2. Material filter: This filter also has the ability to filter all visualizations in the material revenue dashboard and has the same ability to filter multiple selections of materials.
3. Revenue and quantity by month trend bar and line chart: By analyzing this chart, we can compare the revenue (with bar) and quantity (with line) for all materials per month.

4. Total revenue: It shows the total revenue as a reflection of the selected combination. These elements can be filtered by year, material, and material type filters as well as month from the trend bar chart.
5. Average of quantity per order: This visualization shows the average quantity per order, which is affected by the year, material, material type and month selection.
6. Material type filter: There are four types of materials that can be used for comparing the revenue which was made with the sales of OEM products, own products, raw materials and semi-finished products. But only the OEM products and own products are sold which means that it can be only made revenue by these material categories. This filter also has the ability to filter the whole visualization in the material revenue dashboard and has the same ability to filter multiple selections of material types.

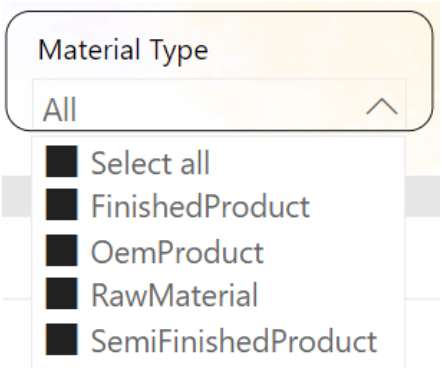
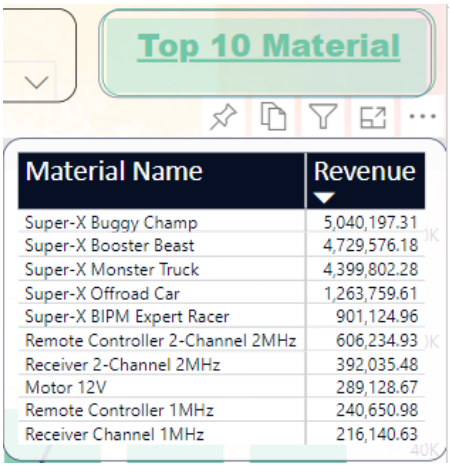



**Figure 90: Material Type filter**

7. Top 10 material button: By clicking the top 10 material button, the dashboard will display a list of the top 10 materials with the highest revenue. This table can be affected by year, material type and month filter selection based on a trend bar and line chart but not by the material filter.




| Material Name | Revenue |
|---|---|
| Super-X Buggy Champ | 5,040,197.31 |
| Super-X Booster Beast | 4,729,576.18 |
| Super-X Monster Truck | 4,399,802.28 |
| Super-X Offroad Car | 1,263,759.61 |
| Super-X BIPM Expert Racer | 901,124.96 |
| Remote Controller 2-Channel 2MHz | 606,234.93 |
| Receiver 2-Channel 2MHz | 392,035.48 |
| Motor 12V | 289,128.67 |
| Remote Controller 1MHz | 240,650.98 |
| Receiver Channel 1MHz | 216,140.63 |

**Figure 91: Top 10 material table**

When analyzing the material revenue dashboard, we found out that the total revenue throughout the years from 2010 to the beginning of 2017 that Super-X achieved is more than 18.212.000 Euro with more than 16.334.000 Euro contributed by sales of the own products. The best performing product regarding the revenue is the Super-X Buggy Champ with more than 5.000.000 Euro. In addition, we found out that in average, five items were ordered with each order.

A last thing to mention is that Power BI has a mobile application which is very convenient for the use of the dashboards. Below is an example of our dashboard on the Power BI mobile app.


**Figure 92: Power BI mobile dashboard**

## 5.3. Evaluation and comparison of the used data warehousing technologies

Our decisions about the used data warehousing technologies were mostly based on our level of knowledge on how to use these tools. In this way, we could assure high quality results on time. Based on our experience, this section delivers an evaluation of the used tool based on criteria like connections, functionalities, error handling, user interface and experience. In the end, a comparison between Tableau and Power BI is included.

**Talend Open Studio for Data Quality**

First of all, Talend Open Studio is an open-source tool and has a very good performance for data quality checking, which makes it preferable to be used from many users.

When it comes to connection, Talend gives us the opportunity to create a connection either to databases or to delimited files. In our case, it is a useful thing because we had csv files to work with. When connecting to a database, Talend is very flexible because we can also connect directly from a catalog or schema.

Regarding functionalities, it is interesting that it offers very general but also really detailed data examinations. We can go from a structural analysis, to a table analysis, to a column one. Moreover, redundancy analyses explore different relationships between different tables. Whenever we want to create a new analysis, by clicking at it, an explanation part will appear, letting us know what results we can get from it. Also, we can choose different indicators, which can make our analysis insightful (for example text statistics, summary statistics, pattern frequency statistics, etc.). Even if we did not make use of it during this project, we would like to add the fact that Talend offers the opportunity to use a SQL editor to create and store SQL queries.

During our work with Talend, we did not face a lot of errors, but whenever an error appears, it is clearly explained at the workspace log what is causing the error. To be mentioned is the fact that ‘Help Content’ from the help menu is useful to check whenever we might experience a difficulty.

Another thing to add is that it has a user-friendly interface. Analyses settings are easy to understand and follow. Thanks to different buttons and selection options, it is very easy for the user to adapt with its functionalities. What makes it even more practical is that we can change analysis parameters within one window, visualize different results without having to repeat previous steps. Analysis results contain charts, descriptive tables and sometimes even a clarification text. It means that the results are clear and easy to understand.

**SQL Power Architect**

Using SQL Power Architect as a tool to make a graphical representation of our galaxy schema was useful for ensuring that database relations and constraints are setup properly. We should mention that our work with Power Architect was limited only on the schema creation which is why our evaluation is mostly based on this.

Starting with connection, we would like to mention that different source connections can be made with different databases. In our case, it was helpful. By connecting Power Architect with superx_development, we could copy and derive our galaxy schema.

Regarding the functionalities, we only used the schema creation and forward engineering SQL script. Functions like performing data profiling on source databases and auto-generation of ETL metadata, were not used during this project. Hence, we cannot give an evaluation about them. The schema creation is easy and fast. By using tools on the left corner, we can add tables or columns, specify their name and type (for columns), specify constraints and create identifying or non- identifying relationships. For a better visualization of our schema, we can click on automatic layout and Power Architect will arrange table orders. Forward Engineering is a quick functionality to transfer and create our schema in the destination database. This function saves a lot of time compared with using SQL commands to create tables.

Error handling seems to be manageable. If there is an error within the script, it will not run and a window with an error explanation will appear. Usually, the explanation is very clear, and we can easily find out what we should change. It is also helpful that in some cases, Power Architect can fix the errors, if we click on QuickFix All option.

About the user interface, Power Architect is easy for people who work with different tools, but it is a little bit old school. Nevertheless, the working field is well organized, with connection in the left, tool on the right and enough space on the middle where the schema is created. Accessing the tools to build tables is easy. Firstly, we have to click on the icon and then click on the area/table where we want to make the changes. It would be better if this process would rather work as drag and drop, as this is a trend for a lot of tools nowadays.

**Pentaho Data Integration**

In this project we used the Pentaho Data Integration community edition which is an open-source tool for the ETL processes. The connection diversity is one of the reasons why companies choose to use Pentaho. It offers different types of database connections like PostgreSQL, Oracle, Snowflake, MS SQL Server and others. Pentaho can also connect with different data sources at the same time and transfer data from one source to another.

About functionalities, we used Pentaho for the ETL processes. Additionally, it is considered to be a good tool for report creations, too. Continuing with the ETL processes, the possibility of extracting data from many sources is very helpful. Moreover, it offers different steps which can be used to clean and transform these data before conducting results. One good quality to mention is the option of doing sample transformations and jobs. When working with too much data and for testing any transformations/jobs, it is possible to either preview the results for a specific selected number of rows or to stop the running instance and restart it to when it was left. This second case is helpful if you also want to see how loaded data look like in the destination database and if there is any problem with the output table (a common mistake is when your source datatype is not the same as the destination datatype).

Despite of all the good quality mentioned, based on our experience, error handling is not a strength of Pentaho. Sometimes the error codes we received were not clear and that is why more explanation is recommended for a quick solution of the problem. The online community and documentations seem to be old and not updated. Also, the database connection is timed out after a period of time, especially if you are working with too much data.

The user interface of Pentaho is very user-friendly and easy to grasp. On the left side of the window, all steps are listed which can be moved to the main window by drag and drop. Also, it is helpful there are descriptions on what a particular step can be used for. Furthermore, the steps are clustered which makes it easy to search and find the right one. It is even suitable for people who do not have very good SQL skills, as SQL code will be automatically integrated. Of course, changes to the script might be needed.

**DBeaver**

DBeaver seems like a great tool to manage different types of databases in a simple way. Actually, it supports a lot of popular database types and you can create different connections in the same time.

When it comes to functionality, the first thing to mention is that exporting and importing data is very easy. Not only this, but also the possibilities of data visualization that are offered by the tool are very good. You can see how all the tables are connected in an ER diagram and you can make changes and modifications directly from there. The option to filter and sort tables from query results can be helpful to get a quick insight about your data. Data manipulation can also be done without the need to write SQL queries. Only by using interface functionalities, you can easily add rows and columns, delete or update them, change the data type, table name or others. Another nice functionality that it offers is inline editing. Autocompletion of SQL code with tables/columns names is also helpful when you are writing queries and working many tables or databases at the same time.

About error handling, in our experience, DBeaver is not the best. It usually identifies where the problem is happening in you code, but for people with little experience, it can be difficult to identify what the exact error is about. Sometimes, due to inactivity, the Pentaho connection to the database is easily to disconnect and asks for establishing the connection again.

When it comes to user interface, it is very flexible. We are able to hide and show list of tools by adapting it based on your preferences. In our opinion, Pentaho has a lot of features and could be useful as an advanced ETL tool.

**Tableau**

Tableau is a very powerful data visualization tool. It comes with various options such as a desktop application or an online version. In our case, we are using the desktop version. One of the best things about Tableau is that it offers a high number of connection options, from CSV files to different servers including Google Ads, Salesforce and others. Users can either choose to extract data or to establish a live connection. Live connections offer a dynamic data display.

Regarding functionalities, we want to mention the ability of Tableau to perform cross-database merges. Instead of being limited to a single data source, you can combine data across departments, create deeper analyses and support the decision making in a better way. Another point worth mentioning is that Tableau's output options include a variety of chart formats as well as mapping capabilities. It means that we can create color-coded maps that display geographically important data in a much more digestible format than tables or charts. Even though dashboards are easy to design, we have to create the charts on a different worksheet first. But after doing so, all charts from the same dashboards are interactive with each other. For example, if we want to filter data from an output chart for a specific year, we just create a filter and then click on the year and it acts as a filter for all the charts within a dashboard.

Based on our experience, we did not have problems with facing errors in Tableau, since most questions are answered within the Tableau online community.

A weak point of Tableau is that it is very confusing for inexperienced users. Most functionalities and options are well hidden, and the user has to search a long time until finding them. Furthermore, it is necessary to have a concept to identify measures and dimensions. Nevertheless, users can quickly adapt to the way Tableau works by exploring the tool and reading through the provided tutorial videos.

**Power BI**

Power BI is also a widely used visualization tool. It offers the opportunity to connect with different data sources and users can easily load and integrate this data.

Evaluating the functionalities, it is possible for Power BI users to create several charts, views and dashboards for different audiences. This dynamic dashboard enables a user to create very detailed reports/dashboards on very specific elements without changing the data source. Another interesting feature is to share a dashboard with other users which is especially useful when it is used within a team or department. Furthermore, Power BI has an alerts/notification ability that allows users to add comments about specific charts and notify other users when an update is available. In addition, users can generate reports from real-time data sources or they can set up daily updates to synchronize the data and update the reports constantly. Another thing worth mentioning is that Power BI supports languages like Dax and M-Query which are similar to VBA and excel formulas. That makes Power BI easier to understand by non-IT users.

When it comes to error handling, Power BI is usually very reliable. The errors are written clearly and can explain solutions. On the other hand, the Power BI community is so large that it is very easy to find a solution whenever users encounter a problem.

In terms of the user interface, it is similar to other Microsoft tools which makes the learning process very fast. The process from not knowing anything about Power BI to creating a reasonable and functioning dashboard takes a much shorter amount of time in comparison to similar products.

**Comparison between Tableau and Power BI**

In general, Tableau and Power BI are powerful tools for data visualization and are easy to use. The common 'drag and drop' functionality offers the opportunity to create dashboards in a very short time. Nonetheless, there are some key differences that we noted.

| Tool / Criteria | tableau | Power BI |
|---|---|---|
| **Connections** | Numerous number of connection | Limited acces to some databases and servers |
| **Functionalities** | Main foucus on data visualization | Main focus on data manipulation |
| **Error Handling** | Large community forum | Limited for free users |
| **User Interface** | Less intuitive features | Easier to learn |

**Figure 93: Main differences between Tableau and Power BI**

- First of all, both tools support multiple connections, but Tableau is winning at the moment. Power BI has limited access to some databases and services which users can easily connect to on Tableau. For some databases, like SAP Hana, Hadoop, or even Postgres, you need to install a connector.
- One important difference we have noticed is referring to the visualization. The two tools offer different approaches. Although Tableau is mainly focused on visuals, Power BI offers data manipulation features first and then provides simple visualization. Users can get answers to their questions simultaneously while analyzing data visualizations on Tableau. It also shows predictions for 'what-if' analyses that hypothetically adjust data to visualize data comparisons dynamically. On the other hand, Power BI provides a rich functionality

such as creating correlations between various data sources. Tableau offers more visualization flexibility but cannot manipulate data as well as Power BI.

- Both tools offer a large community forum where you can address different questions and errors that you might face. But for Power BI, the official community is limited to free users who cannot access all the guidelines offered.
- Tableau is less intuitive than Power BI and users can discover a lot of functions hidden among the menus. For example, a great function that we found while working with Tableau is prediction (forecasting), where users can choose which model to use (linear regression, logarithmic regression etc.). On the other hand, Power BI is easier to learn.

To conclude, there is not a big difference between Tableau and Power BI regarding the features. The selection of what tool to use should be based on the requirements of the user. Power BI is mainly focusing on different stakeholders while Tableau caters mostly to analysts. If you work with a limited amount of data, Power BI will perform better and faster. If you are handling bulky data, Talend is the solution. If data visualization is your focus, Tableau offers some advantages. But if you want to dive deeper into your data to explore, predict or optimize models, Power BI is the way to go.

# 6. Process mining

This chapter is dedicated to the second challenge of this project which is about the analysis of the Sales process and its performance by implementing process mining. As a first step we used the event log table in the operational database to discover the Sales process. Afterwards, we analyzed the discovered process in terms of frequencies and performance. The findings are an addition to the KPI analysis of the previous chapters since they were also used for deriving recommendations for the Super-X management.

## 6.1. Process discovery

To make use of process mining, an event log is needed which provides at least information about case ids, activities and timestamps. Therefore, the table event_logs was used as it includes all the relevant information for creating a model of the Sales process. It had to be dealt with the problem that this table doesn't contain clear case ids that link all activities which belong to one case to each other. As the columns of retailer_id, supplier_id and material_id include Null values and are not used in all the processes, they were not suitable for case IDs. Considering that a case determines the beginning and end of a process and allows a specific perspective on the process, an option is to combine multiple columns instead of using one (Rozinat, 2017). For this reason, we created case IDs which are a combination of the department, the year and the month. This means that all activities which were executed by a department within one month were linked to each other and treated as one case. Usually, this approach would not be suitable as many business processes include several departments and also the activities within one process don't have to be necessarily executed within the frame of a month. Again, the knowledge we gained during the BPM project came in handy as we knew, that in the case of Super-X, all the processes are executed on a monthly basis and the departments work relatively independent from each other. These circumstances allowed us to proceed as described before. Examples of the combined case IDs are shown in the following figure.


**Figure 94: Combined case IDs**

Regarding the tool, we decided to use Disco, provided by Fluxicon, as it has a very user-friendly interface and we were already familiar with it. Disco makes use of the 'Fuzzy Miner' which was developed by Christian W. Günther who is a co-founder of Fluxicon. This algorithm 'uses significance/correlation metrics to interactively simplify the process model at desired level of

abstraction' which enables it to leave out or cluster less important information (Rozinat, ProM Tips — Which Mining Algorithm Should You Use?, 2010).

As a first step, the event_logs table was exported from DBeaver as an csv file and imported to Disco. The columns year_month and department were set as case ID, the information about the activities were taken from the column of activity, the columns start_timestamp and end_timestamp were used as the timestamp. Additional to these mandatory columns, the employee IDs were taken into consideration as a resource and the retailer IDs were set to the category of 'other' as they are not resources. This was done because both columns provide interesting insights into the Sales process. After setting everything up, we were able to do the process discovery with the help of Disco. Since we are focusing on the Sales process, we filtered only for case IDs which include Sales. The following figure shows the process model we retrieved from the event_logs table.

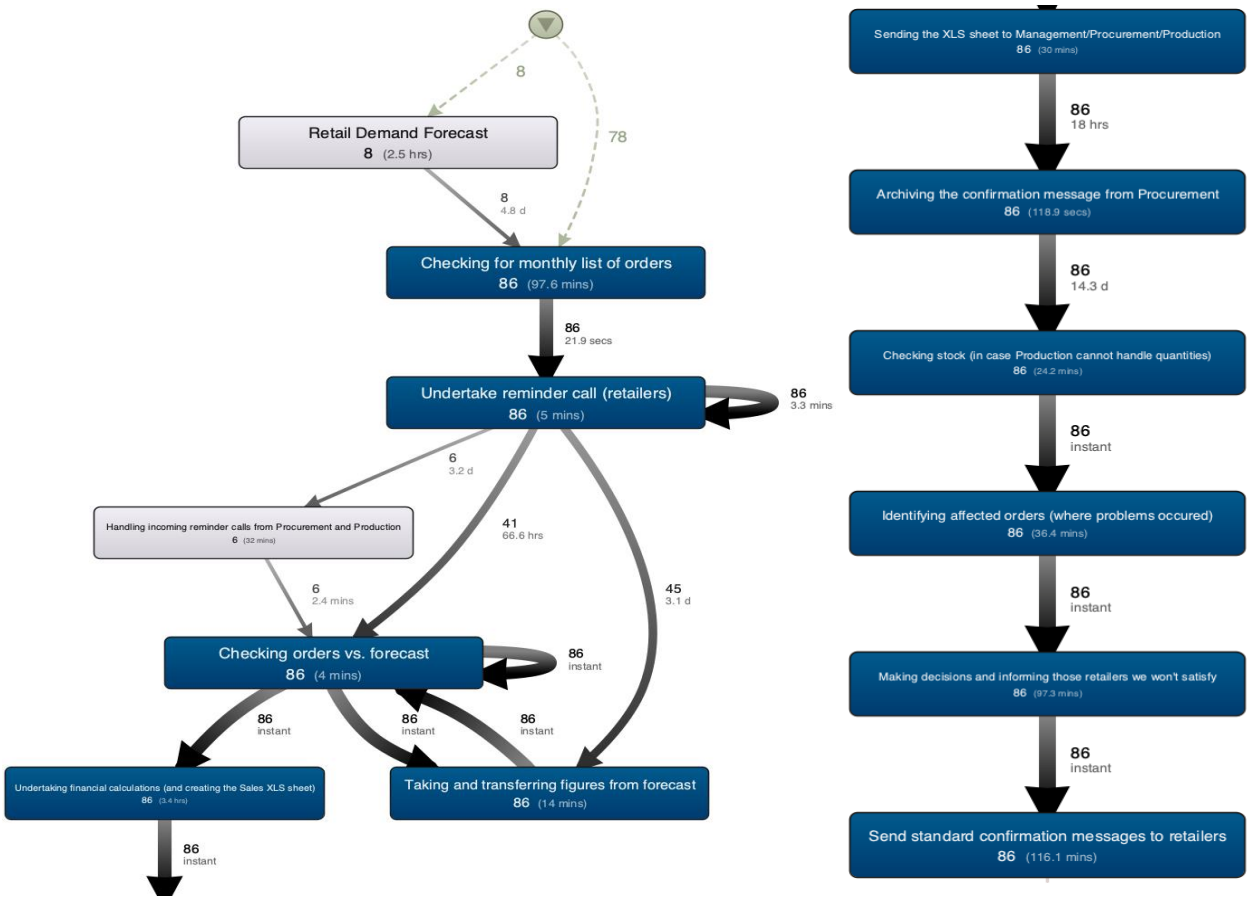



**Figure 95: Mined Super-X Sales process in Disco**

## 6.2. Four competing quality criteria

After mining the process from the event log, the quality of it can be characterized by four different dimensions: fitness, simplicity, generalization and precision (van der Aalst, 2012). These four criteria are competing with each other which means that it is usually not possible to meet all criteria perfectly at the same time (Gehrke & Werner, 2013). In the following sub chapters, the four competing criteria will be applied to estimate the quality of the mined process.

**Fitness**

Fitness describes the level of ability of a model to 'replay all behavior recorded in the event log' (Gehrke & Werner, 2013). Since the process model is mined from and built on the real event log, all behavior that is recorded in the event log can be replayed. Therefore, a perfect level of fitness is determined. But as most of the times in life and even more in data science, there is a tradeoff. As the title of this chapter already states, the perfect level of fitness will compete to other quality criteria.

**Precision**

Precision addresses the level of allowance for 'additional behavior very different from the behavior recorded in the event log' (Gehrke & Werner, 2013). Between the tasks 'Checking orders vs. forecast' and 'Taking and transferring figures from the forecast', a loop can be identified in *Figure 95*. This loop allows an infinite number of executions of the mentioned tasks that lead to cases, that are not present in the event log and therefore describe different behavior. However, the cases recorded in the event log also take a decent amount of the mentioned loop, which means that this behavior is not unusual. Therefore, the mined process does not have a perfect but still a decent level of precision.

**Generalization**

The mentioned tradeoff can be seen for the criteria of generalization. If a 'process model is not exclusively restricted to display the eventually limited record of observed behavior in the event log but that it provides an abstraction and generalizes from individual process instances' (Gehrke

& Werner, 2013), it has a perfect level of generalization. As already described above, the process model might allow some different behavior, but it is still quite precise. It does not show any abstraction or generalization from individual process instances because all process instances named in the event log are also shown in the process model. Even when changing the activity and path percentage in the map view of Disco, the same model is shown. For this reason, a poor level of generalization can be established.

**Simplicity**

Last but very not least for the competing quality criteria is simplicity. It describes 'that the simplest model that can explain the observed behavior should be preferred' (Gehrke & Werner, 2013). Usually, a perfect level of fitness means a complex model with all tasks in the event log and a lot of paths connecting them, which then usually leads to a low level of simplicity. But in this case, a first look on the process model reveals the opposite. Due to the fact that there are not a lot of different tasks in the process and the fuzzy mining undertaken in Disco leads to smart connections of the tasks as well as a rather simple overview, the process model looks relatively simple and lean. Moreover, the activity and path percentage in Disco does not change the model. Therefore, the level of simplicity can be determined as rather high.

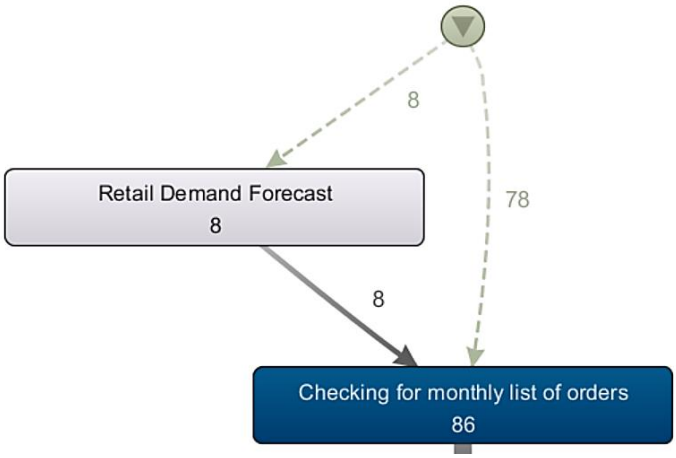
## 6.3. Frequency analysis

The first part of the analysis covers the frequency within the process. Looking at the statistics provided by Disco *(Figure 96*), we found out that in total 86 cases went through the Sales process. Since each case reflects one month, it means that the data cover 86 months which are about seven years. To be more precise, the analyzed time period starts on 1st January 2010 and ends on 24th February 2017. The process contains 13 activities but not all cases went through every activity. In total, the event log documented about 11.500 events regarding the Sales department.

| | |
|---|---|
| Events | 11,499 |
| Cases | 86 |
| Activities | 13 |
| Median case duration | 19.4 d |
| Mean case duration | 19.3 d |
| Start | 01.01.2010 09:00:00 |
| End | 24.02.2017 17:17:50 |

**Figure 96: General statistics of the Sales process**

After getting a general overview, we had a deeper look into the process itself. The first thing that we noticed is the different start event of some cases. Eight of them started with the activity of creating the retailer demand forecast while the rest directly started with checking and monitoring the list of orders. After looking at the case IDs this situation can be easily explained as the forecast is only created in January and is valid for the entire year (see *Figure 97*).




| Case ID |
|---|
| Sales-2010/1 |
| Sales-2011/1 |
| Sales-2012/1 |
| Sales-2013/1 |
| Sales-2014/1 |
| Sales-2015/1 |
| Sales-2016/1 |
| Sales-2017/1 |

**Figure 97: Cases including Retail Demand Forecast**

*Figure 98* reveals a lot of different information regarding the Sales process. In the following, all findings regarding the shown process activities are listed and underlined by additional analyses.

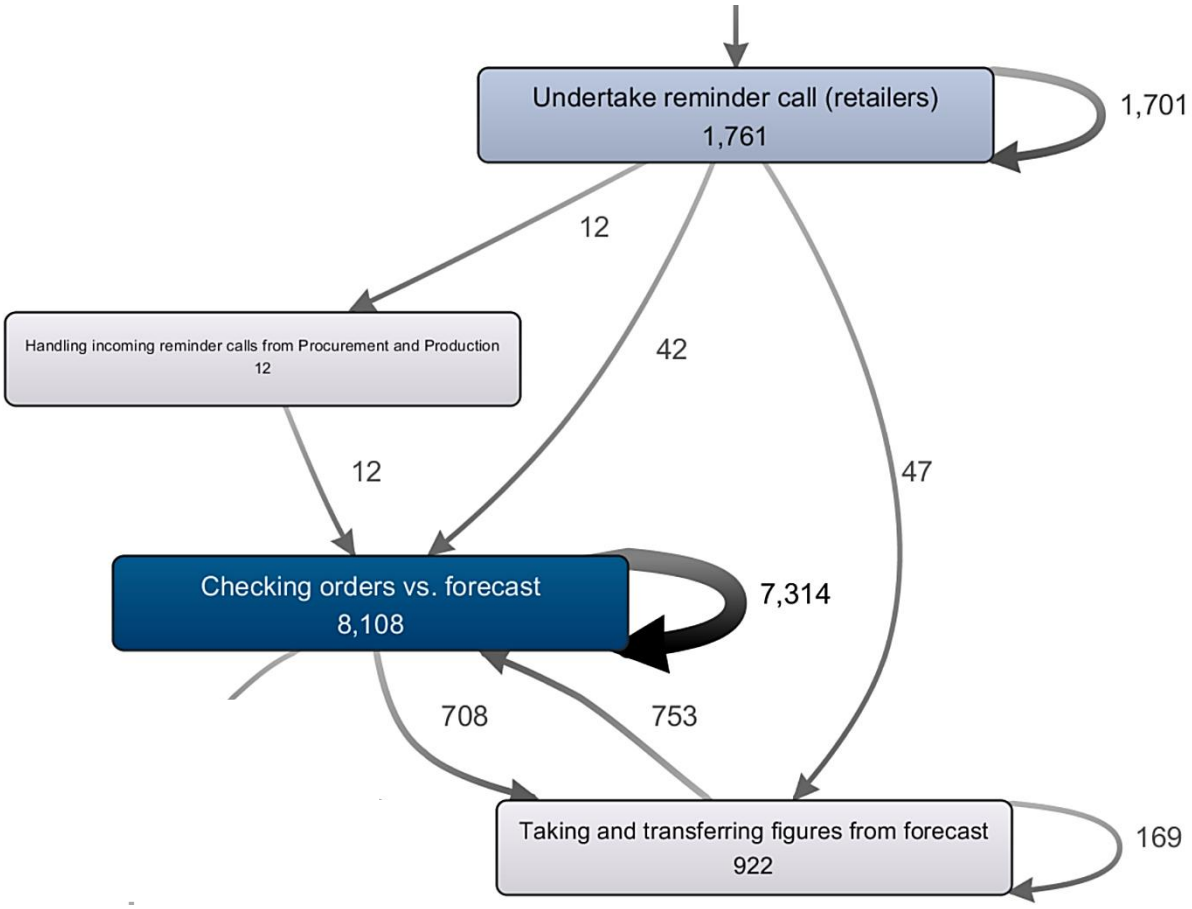



**Figure 98: Absolute frequencies of selected process activities**

One finding is the high number of reminding calls to the retailers which were necessary every month. The frequency shows that 1.761 reminder calls were made during the analyzed 86 months. Since several calls were made during each month, 1.701 repetitions of this activity can be seen in the process model. The high frequency and number of repetitions reveal a problem within the process. An explanation could be that it is not that comfortable and easy to place an order at Super-X. As we know from the BPM project, the retailers must fill in a template and send it via email. Additionally, they are only allowed to order once a month which could lead to the situation that many retailers wait for the last moment to be able to improve the estimations about their demand for the current month. The following table shows all retailers which needed a reminder call and ranks them in terms of how often they received such a call. The table enlists 29 retailers which is about 28% of all retailers (total number is 105, see chapter 3.2). 8 out of

these 29 retailers were reminded in 100% of the cases, which means every month. Another 11 retailers needed a call in more than 70% of the cases. Only one third of the enlisted retailers had to be called by Sales in 50% or less of the cases.

**Table 5: Ranking of retailers that needed at least one reminder call**

| Value | ▲ Frequency |
|---|---|
| 20 | 86 |
| 21 | 86 |
| 29 | 86 |
| 37 | 86 |
| 38 | 86 |
| 45 | 86 |
| 50 | 86 |
| 58 | 86 |
| 49 | 85 |
| 48 | 84 |
| 97 | 84 |
| 32 | 83 |
| 94 | 82 |
| 15 | 79 |
| 44 | 79 |
| 19 | 72 |
| 1 | 70 |
| 25 | 68 |
| 71 | 61 |
| 46 | 45 |
| 13 | 43 |
| 35 | 42 |
| 12 | 21 |
| 77 | 18 |
| 64 | 15 |
| 84 | 13 |
| 39 | 13 |
| 90 | 12 |
| 60 | 4 |

After the activity of undertaking the reminder calls is done, there are three different following activities. The first option is that Sales handles incoming reminder calls from Procurement and Production. This step was executed in only 6 cases (about 7% of all cases) with a total frequency of 12 times (since they receive two calls, one from Production and another one from Procurement). The number is relatively low, but this step could also be eliminated if the orders would be placed on time by the retailers or the internal communication would be improved.

If no reminder call was received by Sales, they either start directly checking the received orders against the forecast (41 cases) or they take and transfer figures from the forecast before the checking (45 cases). The latter case means that the retailer did not place the updated figures on time and therefore, a Sales employee had to manually take and transfer the figures from the forecast. As the sequence of these two steps is interchangeable and it depends on which order is checked first, there is a lot of looping between these two activities. This is caused by the circumstance that the employees check for every retailer if the order was placed on time or not which influences the sequence of the activities. The following table shows the number of delayed orders per retailer. In total 19 out of 105 retailers placed their order too late at least once. Regarding the retailer with the ID number 29, the order was placed too late in 100% of the cases. Furthermore, the retailers with the IDs 20 and 38 had this issue nearly every month. Looking at the absolute frequency, Sales had to deal with 922 delayed orders out of approximately 9.000 orders in total (8.108 + 922 = 9.030)[1], which is about 10%.

**Table 6: Ranking of retailers with at least one delayed order**

| Value | ▲ Frequency |
|---|---|
| 29 | 86 |
| 20 | 83 |
| 38 | 80 |
| 45 | 78 |
| 97 | 76 |
| 94 | 55 |
| 15 | 48 |
| 48 | 47 |
| 1 | 45 |
| 21 | 44 |
| 32 | 43 |
| 49 | 43 |
| 25 | 42 |
| 44 | 42 |
| 50 | 42 |
| 19 | 41 |
| 71 | 23 |
| 46 | 3 |
| 12 | 1 |

[1] The orders table in the database contains 9.111 orders in total which means that there is a slight difference. A reason for this could be that some orders were canceled before they were checked against the forecast.

A last problem which was identified during the frequency analysis is shown in *Figure 99*. This part of the process model reveals that not all demands of Sales could be entirely met by Production every month (100% of the cases). This has the consequence of not satisfying all customers which can have a big impact on the reliability and the image of Super-X. Unfortunately, the number of affected orders and retailers cannot be analyses as these pieces of information are missing in the event log. But in general, this situation inflicts that there is a problem within the production department as Super-X is never capable of satisfying all customers.

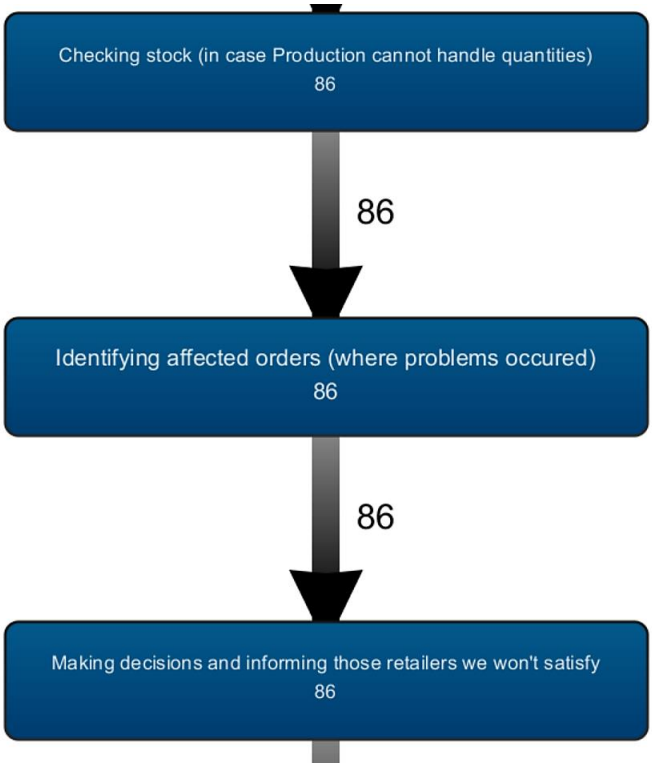



**Figure 99: Cases with unsatisfied retailers**

## 6.4. Performance analysis

The most interesting and insightful part of analyzing a mined process is to see how the process of the corresponding company, department or other organizational unit is performing. Since we are living in a world that is getting faster and faster, time management becomes increasingly important. But there is an even more crucial factor to be considered when it comes to performance, which is money. Since time is money and most organizations obviously want to

save it to be more profitable, an effective way to analyze the performance of a process is to look at different perspectives of time in it.

The first focus will be on the mean times of tasks and paths. The following *Figure 100* will help to understand and visualize the upcoming discussions. The darker the color of the task or path is, the longer is the mean duration of it. The mean duration is also shown within the tasks and besides the paths.

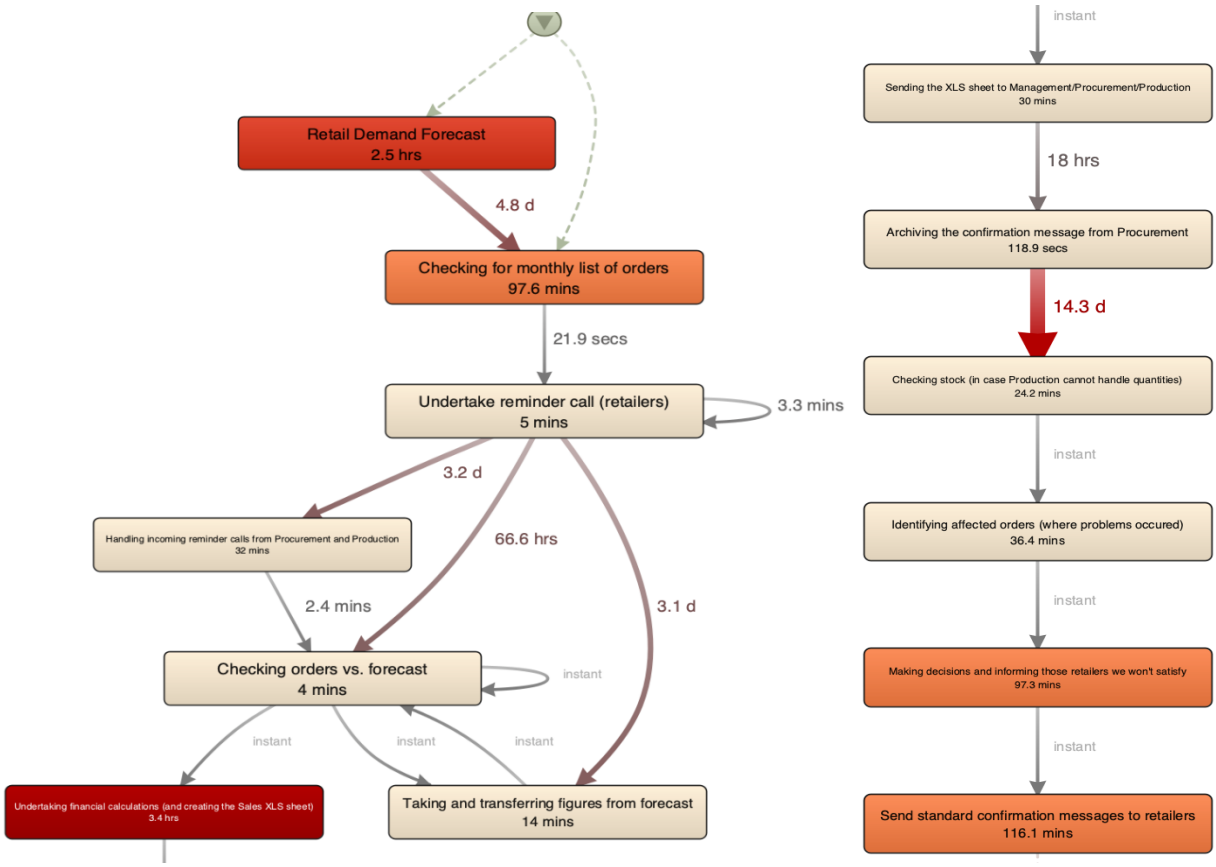



**Figure 100: Mined Super-X sales process in Disco (mean duration)**

A first look reveals five bottlenecks: 'Retail Demand Forecast', 'checking for monthly list of orders', 'undertaking financial calculations (and creating XLS sheet)', 'Making decisions and informing those retailers we will not satisfy' as well as 'Send standard confirmation messages to retailers'. These tasks slow down the process in comparison to the mean duration of other tasks

and cannot be bypassed. Therefore, it is worth to have a closer look into all the bottlenecks to analyze reasons.

Starting from the top, 'Retail Demand Forecast' is, as already described in the *Frequency* analysis, a task that is only done once a year in January to forecast retailer orders in order to plan the upcoming year. Therefore, this task has a higher workload than the other tasks that are done monthly. For this reason, it is absolutely reasonable that 'Retail Demand Forecast' has a mean duration of 2.5 hours and its outgoing path takes 4.8 days on average. This extra task in January is reflected in *Figure 101*. It shows the months of the event log ordered by duration. The leading case ids are all records from January.

| Case ID | Events | ▲ Duration |
|---|---|---|
| 2016/1 | 133 | 25 days, 11 hours |
| 2015/1 | 138 | 25 days, 9 hours |
| 2010/1 | 139 | 25 days, 8 hours |
| 2014/1 | 136 | 23 days, 11 hours |
| 2013/1 | 134 | 23 days, 10 hours |
| 2017/1 | 135 | 22 days, 12 hours |
| 2011/1 | 133 | 22 days, 10 hours |
| 2012/1 | 134 | 22 days, 7 hours |
| 2011/12 | 129 | 19 days, 14 hours |
| 2013/4 | 133 | 19 days, 14 hours |
| 2011/8 | 133 | 19 days, 14 hours |
| 2014/6 | 135 | 19 days, 13 hours |
| 2011/5 | 132 | 19 days, 13 hours |
| 2010/11 | 133 | 19 days, 13 hours |

**Figure 101: Months ordered by duration**

The initial first task for all months, except January, is checking for monthly list of orders. For just checking the orders, this task takes rather long with a mean duration of 97.6 minutes. As known from the BPM process description, the reason for it is that retailers do not use the prepared template for it. With 3.4 hours on average, undertaking financial calculations (and creating the Sales XLS sheet) has the longest mean duration. This can be attributed to a lot of manual calculations done within XLS sheets.

The last two tasks of the mined sales process are shown in a deep orange color, which indicates a rather long mean duration. Sending standard confirmation messages to retailers is a task that is expected to be done standardized within a short period of time. However, the mean duration

of it is 116.1 minutes. In the interview with the head of sales in BPM it was said that this process takes a long time due to the odd system. For this reason, there is also high potential to shorten this task and with it the whole process by implementing smart software solutions for example.

As also known from the BPM process description, making decisions and informing those retailers we won't satisfy, has a rather high average duration because the decision making is based on personal decisions and historical relations to the retailers. So, it might be reasonable to take a bit more time on this task. Nevertheless, an optimization of the whole Super-X process would lead to a higher satisfaction of retailer needs, which would shorten the mean duration of this task dramatically, since almost all retailer demands can be met.

Thinking about the whole Super-X process leads to a closer look on the path duration. More specifically on the path between archiving the confirmation message from procurement and checking stock (in case production cannot handle quantities). Not only the color but also the mean duration itself (14.3 days) state that production and procurement take very long to identify if all retailer orders can be met. This path is by far the longest part of the whole Sales process. For this reason, it will bring high value to the process performance analysis to check the performance of the production process. Since this is not in the scope of this project, no further investigations on this direction will be undertaken.

The second focus will be on the total duration of all tasks and paths. A visualization similar to the mean duration can be found in *Figure 102* below. Let's first have a look on the already identified and discussed bottlenecks, how the visualization of them changed and what that means.

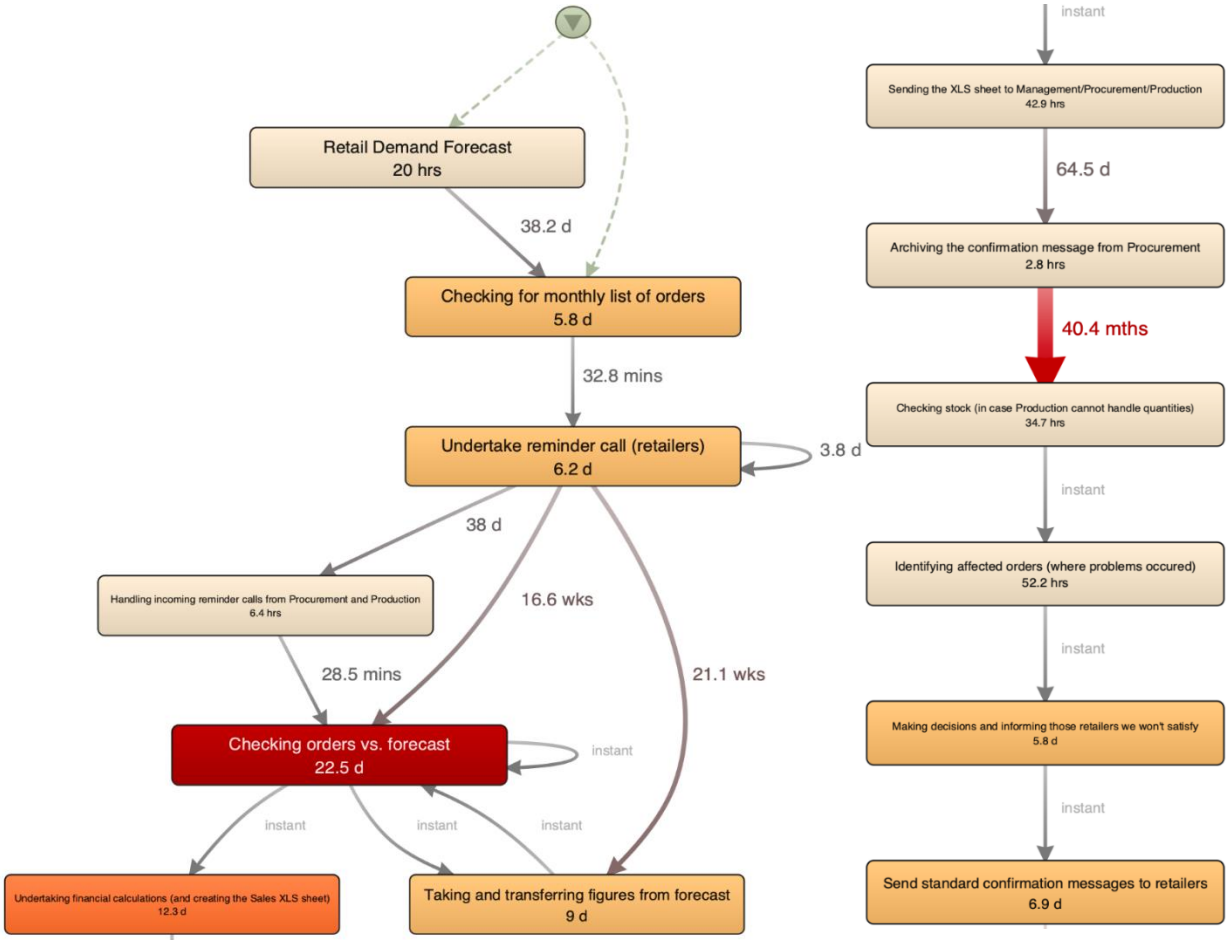



**Figure 102: Mined Super-X Sales process (total duration)**

'Retail demand forecast' is not shown in dark orange anymore, which proves the theory that the rather long mean duration of this task is reasonable in comparison to the whole process, since this task is only done once a year. This theory can be proven by *Figure 103*. 'Retail demand forecast' has a rather high mean duration but is listed on the very bottom of the table, since it has the lowest frequency of all tasks. 'Checking for monthly list of orders' and 'undertaking calculations (and creating sales XLS sheet)' are still highlighted. Even if the colors changed from deep red to orange, it is reasonable to shorten these tasks. Also, the last two tasks of the process remain in the color for a rather long duration. Therefore, the already stated analysis remains the same. Lastly, the path between 'archiving the confirmation message from procurement' and 'checking stock (in case production cannot handle quantities)' still has the longest duration in the whole Sales process.

But there are not only similarities when comparing the mean duration to the total duration. The task that takes the longest with 22.5 days in the view for the total duration now is ‘checking orders vs. forecasts’, which was a rather short task with 4 minutes in the view of mean duration. The reason for this opposite visualization can be found when looking at the frequency of this task. *Figure 103* shows that checking orders vs. forecast has the second shortest mean duration, but by far the highest frequency with a number of 8.108. This means that 71.04% of all tasks done were ‘checking orders vs. forecast’. For this reason, the most time of all activities related to Sales in the event log was used to check orders against forecasts. Therefore, it will also be very valuable to think about solutions to shorten the time needed for checking orders against the forecast.

| Activity | ▲ Frequency | Relative frequency | Median duration | Mean duration |
|---|---|---|---|---|
| Checking orders vs. forecast | 8,108 | 70.51 % | 4 mins | 4 mins |
| Undertake reminder call (retailers) | 1,761 | 15.31 % | 5 mins, 3 secs | 5 mins, 1 sec |
| Taking and transferring figures from ... | 922 | 8.02 % | 14 mins | 14 mins |
| Checking for monthly list of orders | 86 | 0.75 % | 1 hour, 34 mins | 1 hour, 37 mins |
| Undertaking financial calculations (a... | 86 | 0.75 % | 3 hours, 37 mins | 3 hours, 25 mins |
| Sending the XLS sheet to Managem... | 86 | 0.75 % | 30 mins, 24 secs | 29 mins, 57 secs |
| Archiving the confirmation message ... | 86 | 0.75 % | 1 min, 57 secs | 1 min, 58 secs |
| Checking stock (in case Production ... | 86 | 0.75 % | 24 mins, 10 secs | 24 mins, 14 secs |
| Identifying affected orders (where pr... | 86 | 0.75 % | 35 mins, 16 secs | 36 mins, 26 secs |
| Making decisions and informing tho... | 86 | 0.75 % | 1 hour, 34 mins | 1 hour, 37 mins |
| Send standard confirmation messag... | 86 | 0.75 % | 1 hour, 56 mins | 1 hour, 56 mins |
| Handling incoming reminder calls fr... | 12 | 0.1 % | 30 mins, 20 secs | 31 mins, 57 secs |
| Retail Demand Forecast | 8 | 0.07 % | 2 hours, 28 mins | 2 hours, 30 mins |

**Figure 103: Tasks ordered by frequency**

The same analysis can be seen for ‘undertake reminder call (retailers)’ as well as ‘taking and transferring figures from forecast’ on position 2 and 3 in *Figure 103*. With rather short mean durations and a high number of frequencies, these tasks were not analyzed for the mean duration, but have to be considered for the total duration. Both tasks relate to a late or even a missed submission of orders by the retailers.

The last focus in the performance analysis will be on the mean duration of the Sales process in total. As shown in *Figure 96* it is 19.4 days, which means that it takes almost 20 days to just confirm to the retailers if their orders can be delivered or not. This demonstrates all the identified problems and that there is a high need for improvements. As mentioned in the beginning of this

chapter, we're living in a fast-moving world where this mean duration of the whole Sales process is not competitive. But the constantly moving environment also brings chances like this project.

# 7. Business recommendations

Throughout this project we worked a lot on the data provided by Super-X. We studied it to create a dimensional model, analyzed the quality of it, transformed it in the ETL process, applied process mining and in the end, we even created dashboards with key performance indicators. This means we got a well understanding of the Sales department of Super-X and not only created a data mart, but also found opportunities for improvement that we would like to outline in this chapter.

As seen in *Figure 71* and described in the chapter ' *KPIs* visualization', Super-X has very significant high- and low-season which could be related to Christmas and summer vacations. However, based on the fact that in low-season way less employees are needed, it might be a business recommendation to hire contract workers for the high-season. This would mean that salaries would be saved throughout the low-season and employees might not need to do extra hours in the high season.

Another recommendation from the *KPIs* visualization can be derived from *Figure 73.* It shows the sales of the Super-X products per year from 2010 to 2017. It can be seen that the sales of the Super-X Booster Beast or the Super-X Buggy Champ are highly increasing over the years shown, whereas the sales of the Super-X Offroad Car and the Super-X BIPM Expert Racer are decreasing. Therefore, it might be smart to give a facelift to products like the Super-X BIPM Racer and the Super-X Offroad Car in order to make them more popular again, which would lead to an increase in the number of sales and revenue. Another option would be to remove them from the product range but this would need further investigation on the profits made and other factors.

*Figure 78* shows that currently there is an overall cancellation rate of almost 30% that leads to a very high sales loss that even increased over the past few years, which can be seen in *Figure 70*.

Moreover, *Figure 83* reveals that a decent number of retailers have a cancellation rate of around 80%, which should not be acceptable. This could have two reasons. Whether there is a reason for that on the side of Super-X that should be taken care of or there is a problem on the side of those retailers. Should this be the case, it would be a business recommendation to end the business with those retailers, because orders that are cancelled cause some costs in the beginning that will not be rewarded with the payment of this order in the end. Therefore, retailers that have a cancellation of around 80% create probably more costs than revenue.

As known from the chapter '*Process mining'* the Sales process has some activities that take very long. In the following, we will present some business recommendations to shorten those activities. Firstly, the checking of retailer orders takes rather long, because the retailers are not using a specific template that is given by Super-X. This can be avoided by implementing a web interface where retailers can place their orders. This would not only lead to consistent retailer orders to check, but also to fewer spelling mistakes by introducing dropdown menus with given entries. This refers to the problem that in the analysis of the extra orders a lot of spelling mistakes were identified.

Another business recommendation is to integrate the data from the before mentioned retailer order web interface directly into the database. This would not only remove the whole task of checking the retailer orders and transferring them to an excel file, but it would also automate the checking of the orders against the forecast and the financial calculations that are currently undertaken manually and take the major part of the Sales process as described in the chapter ' *Performance* analysis'. So, considering the technological possibilities nowadays, the integration of a system to automate the beforementioned tasks is appropriate and will be a huge time and cost saver.

Writing about technological possibilities, there is one more to mention here. Reminder calls also take a decent amount of time in the Sales process. Due to the fact that most of the communication today is done via e-mail, it is another business recommendation to implement a

mail program that can send a standardized reminder email to all selected retailers at the same time. This again, would save time and related costs.

## 8. Project reflection

The two challenges in this project were on the one hand to design and implement a data mart for the Sales department of Super-X and on the other hand to apply process mining to get more insights into the Sales process. The responsibilities of each team member can be found in the appendix, section B.

A major part of the project was dedicated to the data mart creation and therefore, most time of the project was spent on this aspect. A lot of ideas regarding the business requirements were gathered in the beginning. At the first sight, it seemed not that difficult to derive the relevant dimensions for the data mart from the requirements and to create a concept. The first concept included only one fact table. But then we got aware of the problem that we could not connect each dimension table with the fact table because there were various levels of granularity used. That is why we had to adjust our concept several times until we decided to go with two fact tables in the end.

The next big challenge was the ETL process. One major problem was the data quality of the extra csv files which we decided to include in our fact table. A lot of additional reading was necessary to deal with the quality issues. The second difficulty was to transfer the different currencies of the sold order items to Euro. Since the data covered several years, we did not want to use a fixed exchange rate. A way was found to use the real time exchange rates from a free API web service. To execute the transformation and to load the data to the data mart, a lot of computational power was needed which took several hours to complete the process. In the end, everything worked fine and we were able to realize our ideas.

A remarkably interesting part of the project was the data visualization since we were able to make use of all our previous work and answer the questions we had in the very beginning. For us, it felt like a big achievement which we can be proud on. The use of two different technologies for this part was also very interesting since we could directly compare the functionalities. Whereas it was much easier for example to connect Tableau to the database, it was a bigger challenge to set up a connection in Power BI. On the other hand, a lot of useful functionalities are well hidden in Tableau which makes it less user-friendly than Power BI. In some cases, it was even nearly impossible to change the formatting in the way we wanted. Despite that, both tools are immensely powerful and have a lot of useful functionalities which enabled us to create highly informative and interesting dashboards.

Another tool, which was great to work with, was Disco. After a suitable case ID was created, we got a lot of interesting insights into the data. The tool is very intuitive and has a lot of powerful functionalities. Applying process mining was a lot of fun and the findings were a powerful addition to the findings from the data mart.

Even though, there were a lot of different challenges in this project, it was a fantastic way of applying everything we have learned during the lecture and using the different tools. It helped to understand the connection between each topic and we deepened our knowledge. We were able to help each other out, when someone was facing a challenge in his or her task via an agile workspace in Microsoft Teams and in our weekly meeting that we held every Sunday at 10 AM. In this meeting, we also split up tasks for the upcoming week and showed each other what anyone was working on during the past week in order to make everyone understand everything that was done throughout the project. In the end, it was great to see the realization of our ideas with each step of the project which was based on good teamwork and a lot of effort from every team member.

## References

Ballard, C., Farrell, D., Gupta, A., Mazuela, C., & Vohnik, S. (2006). *Dimensional Modeling: In a Business Intelligence Environment.* USA: IBM Redbooks.

Gehrke, N., & Werner, M. (2013). Process Mining. *WISU – Wirtschaft und Studium*, pp. 934 - 943.

Ges. (2020). *Data*. Retrieved from www.datawarehousing.com

Rozinat, A. (2010, October 18). *ProM Tips — Which Mining Algorithm Should You Use?* Retrieved from Flux Capacitor: https://fluxicon.com/blog/2010/10/prom-tips-mining-algorithm/

Rozinat, A. (2017, August 24). *Combining Multiple Columns as Case ID.* Retrieved from Flux Capacitor: https://fluxicon.com/blog/2017/08/combining-multiple-columns-as-case-id/

Talend. (2017, October 26). *Talend Open Studio for Data Quality: User Guide.*

*United Domains*. (2021, January 10). Retrieved from https://www.united-domains.de/name-domain/

van der Aalst, W. (2012). *Process Mining Manifesto.* Berlin, Heidelberg: Springer.

# Appendix

## Section A

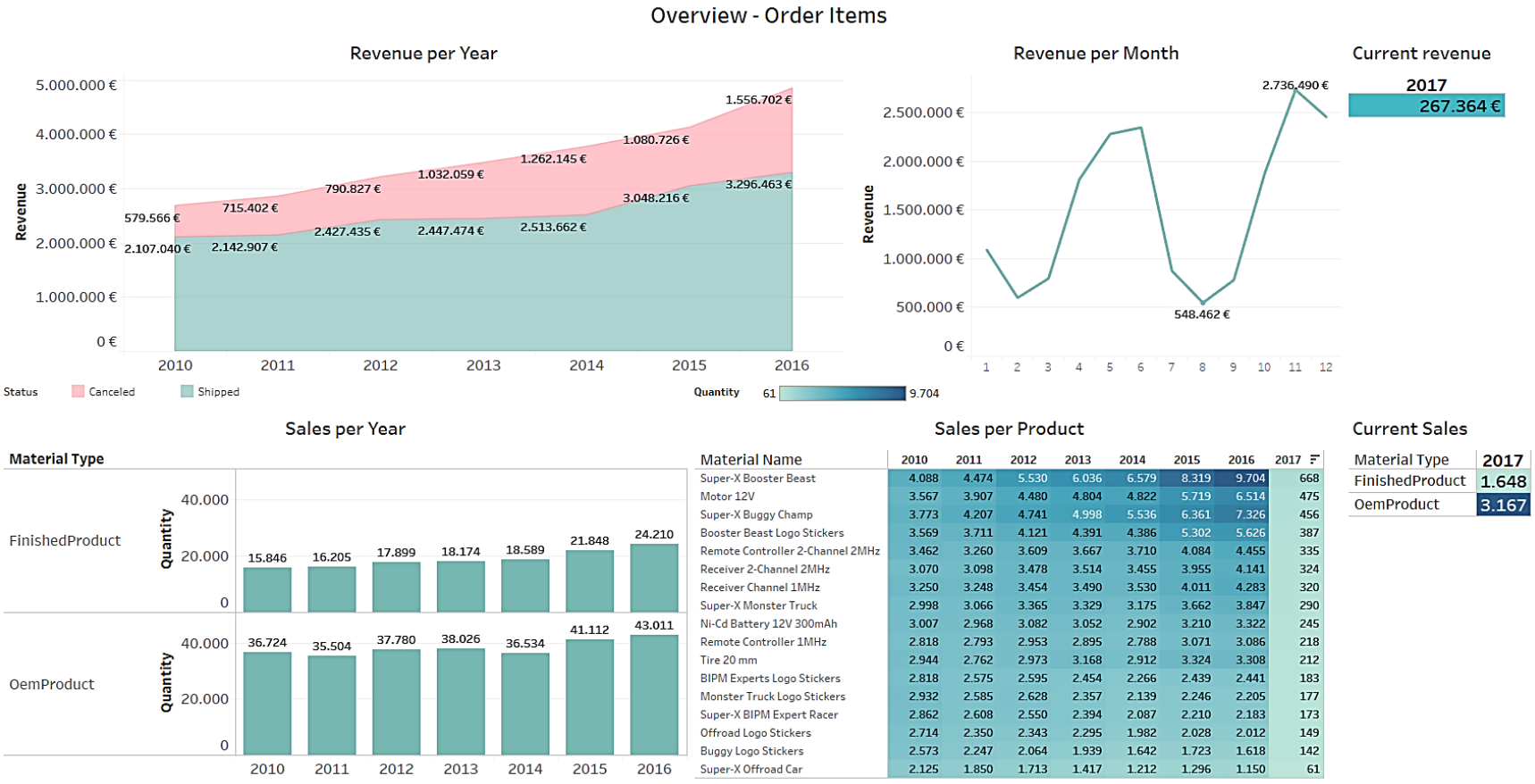



**Figure 104: Dashboard - Order Items**