Lappeenranta-Lahti University of Technology
School of Engineering Science
Degree Program in Computer Science

Master's Thesis

Arttu Tolvanen

DEVELOPING AN EFFICIENT BUSINESS INTELLIGENCE SOLUTION FOR DAY-TO-DAY SUPPLY CHAIN MANAGEMENT: CASE PULP AND PAPER INDUSTRY WORKSHOP

Examiners: D.Sc. (Tech.) Ari Happonen
Professor Jari Porras

Supervisors: D.Sc. (Tech.) Ari Happonen
M.Sc. (Tech.) Eerik Honkiniemi
M.Sc. (Tech.) Eija Kylliäinen

ABSTRACT

Lappeenranta-Lahti University of Technology
School of Engineering Science
Degree Program in Computer Science

Arttu Tolvanen

Developing an efficient business intelligence solution for day-to-day supply chain management: Case pulp and paper industry workshop

Master's Thesis

2019

66 pages, 20 figures, 5 tables, 1 appendix

Examiners: D.Sc. (Tech.) Ari Happonen
Prof. Jari Porras

Keywords: business intelligence, power bi, data modeling, optimization

Business intelligence services offer valuable ways to process and analyze the wealth of data available to companies today. In this thesis, Microsoft Power BI was used to develop a business intelligence solution to assist in the daily operative functions of the procurement processes of a pulp and paper industry workshop. To provide theoretical context for how to develop a business intelligence solution from multiple data sources, key data modeling principles are introduced. The results of the development process are showcased in the empirical section of this thesis, with examples of how the dashboards in the solution can be used to guide daily tasks. Being a tool intended for daily use, its technical performance is important for a successful adoption by the end users. The performance aspects of the newly developed tool were analyzed and optimized based on the findings of the literature review, and the tool was determined to function efficiently.

TIIVISTELMÄ

Lappeenrannan-Lahden teknillinen yliopisto
School of Engineering Science
Degree Program in Computer Science

Arttu Tolvanen

Tehokkaan business intelligence -ratkaisun kehittäminen päivittäiseen toimitusketjun hallintaan: Case sellu- ja paperiteollisuuden konepajayritys

Diplomityö

2019

66 sivua, 20 kuvaa, 5 taulukkoa, 1 liite

Työn tarkastajat: TkT Ari Happonen
Prof. Jari Porras

Hakusanat: liiketoimintatieto, power bi, tietomallinnus, optimisaatio
Keywords: business intelligence, power bi, data modeling, optimization

Business intelligence -palvelut tarjoavat yrityksille hyödyllisiä keinoja nykypäivän suurten datamäärien prosessointiin ja analysointiin. Tässä diplomityössä kehitettiin Microsoft Power BI:llä business intelligence ratkaisu tukemaan sellu- ja paperiteollisuuden konepajayrityksen hankintaprosessien päivittäistoimintaa. Tietomallinnuksen keskeisiä periaatteita esitellään työssä tarjoamaan teoreettista kontekstia business intelligence työkalun kehitystavalle usean datalähteen pohjalta. Kehitysprosessin tuloksia esitellään työn empiirisessä osuudessa esimerkeillä kuinka ratkaisun dashboardit tukevat päivittäisten tehtävien suorittamista. Päivittäiskäyttöön tarkoitetun työkalun tekninen suorituskyky on tärkeä, jotta loppukäyttäjät omaksuvat sen. Kehitetyn työkalun suorituskykyä analysoitiin ja optimoitiin kirjallisuuskatsauksessa esiteltyjen keinojen avulla, ja se osoittautui tehokkaaksi.

PREFACE

In many ways, writing this master's thesis mirrored the half marathon I participated in during the writing process: a steady slog for the bulk of the performance with a frantic sprint to the finish in a last-ditch effort to finish within the deadline for the event. The half marathon did not end well: the final sprint proved to be too much in the sweltering heat, and I collapsed less than a kilometer from the finish line, unable to continue. Procrastination has always been a character flaw of mine, especially in my academic efforts, and deadlines are my most powerful motivator. However, my failure at the half marathon was a valuable lesson: if you cut it too close, sometimes you get burned. This served as a wake-up call for writing this master's thesis, and thankfully I was able to finish it in time.

To make this all possible, I received plenty of help and guidance from the supervisors and examiners of this thesis as well as multiple other people, both from LUT University and ANDRITZ. Many thanks to Jari Porras and Ari Happonen for accommodating my ridiculously frantic schedule. Special thanks to Ari for all the guidance and encouragement as well as getting me in touch with the fine people at ANDRITZ Savonlinna Works Oy in the first place, making this all possible. A very special thank-you to Eerik Honkiniemi and Eija Kylliäinen, the people who started this all and worked with me tirelessly throughout the project. Thank you to Jarmo Pohjolainen for facilitating communications with the IT department and clearing technical hurdles impeding progress, and to Wayna Moncada for providing insightful and valuable feedback throughout the development process. Thank you to Inka Vilpola and Tom Kostiainen for providing me with the tools to start development and providing excellent assistance in technical matters throughout the whole process. It has been a pleasure working with you all for these past 6 months, and I look forward to working with you all in the future.

Thanks to my parents and my sisters for providing me with moral support through all these years of being a student, and to my friends for providing me with immoral support. As my time in Lappeenranta comes to an end, I will begin a new chapter of my life in the metropolitan area. I thoroughly enjoyed my time here, and I'd like to thank LUT University for being a great place to study, LOAS for providing me with excellent housing and a fast Internet connection, and the city of Lappeenranta for being a beautiful place to live in.

TABLE OF CONTENTS

LIST OF SYMBOLS AND ABBREVIATIONS

BI	Business Intelligence
CRM	Customer Relationship Management
DAX	Data Analysis Expressions
ETL	Extract, Transform, and Load
ERP	Enterprise Resource Planning
GUI	Graphical User Interface
IoT	Internet of Things
KPI	Key Performance Indicator
ODBC	Open Database Connectivity
OLE DB	Object Linking and Embedding, Database
OLTP	Online Transactional Processing
OTP	On-time performance
SME	Small and Medium Enterprises
SQDCM	Safety, Quality, Delivery, Cost and Morale
SRT	System Response Time
SSIS	SQL Server Integration Services

1 INTRODUCTION

This introductory chapter covers the background of the subject matter through a brief analysis of the amount of research being done in the field of business intelligence (BI) and analytics as well as the value BI provides to organizations, the goals and delimitations for the thesis work, and the structure of the document.

1.1 Background

Business intelligence systems are becoming increasingly important for many organizations and businesses as the volume of data gathered from their operative processes, environment, and customers grows. Businesses have more data available to support their decision-making processes than ever: the digitalization trend and the emergence of new technologies such as IoT (Internet of Things) systems being utilized alongside more traditional ERP (Enterprise resource planning) applications has led to a rapid increase in the amount of data being gathered, culminating in the term 'big data'. [1] This is reflected in the activity of research being conducted in the field: there has been a strong growing trend of new research publications related to business intelligence in the past 5 years, especially when combined with the 'big data' keyword. [2]

This large and continuously growing amount and complexity of data being gathered is not valuable by itself. People have a limited capability to process information. Therefore, it is necessary to develop tools that assist in making the relevant information available to those who need it for analysis and enable users to construct their own dashboards and analysis tools in a self-service business intelligence environment without expert assistance [3]. According to a survey conducted by IBM Institute for Business Value and MIT Sloan Management Review in 2011, 58% out of more than 4500 respondents reported gaining competitive value through business intelligence and analytics [4]. As business intelligence and analytics remained a high priority for IT spending according to a survey by Gartner in 2013, it can be said that more businesses are beginning to see the value provided by BI solutions [4].

Developing a business intelligence solution requires combining large amounts of data from multiple sources into a single data set. There are many challenges involved in designing such

a data set in an efficient manner. The usage of the word ‘efficiency’ in this work is intended to convey factors that affect performance, such as load times, memory usage, and storage size. Performance can affect user experience, a key factor in increasing the value created by a business intelligence solution used in day-to-day operations by operative personnel, such as the one developed as part of this thesis. In this case, the adoption rate of the new BI solution becomes a factor in its value creation. Experiencing negative emotions or high task loads while using a new technology has been linked to unsuccessful technology adoptions [5]; for instance, high load times can lead to accomplishing tasks with the new technology taking longer than it should, resulting in the users becoming frustrated, and decreasing the likelihood of a successful adoption. Therefore, optimizing performance increases the value provided by the business intelligence solution in this case.

The business intelligence services of today offer a wide variety of interactive data visualization tools. This increased interactivity in turn enables more proactive analysis instead of simply viewing retrospective key performance indicators (KPIs) such as on-time performance (OTP). One such service is Microsoft Power BI [6]. Originally launched in 2014 as a part of Microsoft’s Office365 service, it has since been re-released as a standalone platform and has grown to be one of the most popular BI services available today [7, pp. 6-8]. Other market leaders that offer similar features include Tableau Software [8], SAP SE and their Lumira [9] and Analytics Cloud [10] platforms, and Qlik’s [11] QlikView and Qlik Sense products [12] [13].

The business intelligence solution developed during this thesis work was made using Microsoft Power BI Desktop [14], the free software application used to create and publish Power BI reports. Power BI was uniquely suited as the development tool of choice for this project, because the company uses a wide array of other Microsoft products, such as SharePoint, in their day-to-day operations. As part of the Microsoft Data Platform, Power BI offers great interconnectivity and embedding features for other Microsoft services [7, pp. 9-15, 20-21], and given the self-service BI nature of the thesis project, Power BI’s wealth of easy-to-use data connectors and simple user interface made it the best choice in this scenario [7, pp. 133-137].

1.2 Goals and delimitations

The main goal of this thesis is to develop a business intelligence solution for the procurement department at ANDRITZ Savonlinna Works Oy for the purposes of supply chain management. The first step of implementing such a tool is to build a data model from the various data sources utilized by the procurement department's processes. This is the focus of the literature review portion of the thesis: how to combine data from multiple sources and organize it in an efficient way to enable business intelligence development.

The research questions posed by this thesis are as follows:

- What are the technical approaches and design choices that could be applied to building an effective data model for a business intelligence solution from multiple data sources?
- What are the most significant factors that affect the efficiency of a business intelligence solution from a technical performance standpoint?

The selection of the development tool used in this thesis was made by the IT department at ANDRITZ Oy to ensure alignment with company strategy and business needs. As such, the data model design and optimization techniques discussed in this work will be based on the unique characteristics of the Power BI service. The validation testing of the developed BI solution will be performed after the thesis has concluded, and therefore no end user feedback will be included in the analysis of the results.

1.3 Structure of the thesis

The second chapter of the thesis is a literature review covering the key factors in designing a tabular data set for a business intelligence solution, how to expand it using Data Analysis Expressions (DAX) calculations, as well as the techniques used in optimizing a data model and its calculated fields and measures. Chapter three describes the design and implementation of the BI solution developed for ANDRITZ Savonlinna Works Oy as a part of the thesis. The fourth chapter presents the steps taken to optimize the performance of the implemented BI solution, and the results of the optimization. Chapter five contains discussion about the thesis as well as contemplation on lessons learned during the writing process. Finally, chapter six is the conclusion of the thesis.

2 LITERATURE REVIEW

In this chapter of the thesis, the key terminology and design choices related to building a data model for a business intelligence solution and the factors affecting its performance will be identified and explained through a literature review.

2.1 Data warehousing versus the self-service BI approach

Traditionally, business intelligence systems have been built on data warehouses. However, in today's landscape of BI services that are more accessible than ever before, business users can create reports on their own using tools such as Power BI Desktop with little to no assistance from BI specialists. This has led to the emergence of the self-service BI approach, which is especially suited for small and medium enterprises (SME) that do not yet have a data warehouse or fully-developed BI strategy. [15]

Data warehousing involves retrieving data from source systems, consolidating and cleaning it, and storing it in a dimensional or normalized data store. The data is then periodically updated, typically in batches, and can be queried for use in analysis in business intelligence or other systems. Data warehouses generally hold several years of history. [16, pp. 1-16] A data warehouse is intended to serve as the single source of truth for the business: it is meant to encompass all of the systems necessary for business intelligence, and the data has been cleaned and transformed to be ready for analytics usage, leaving no room for error [17]. While reaching the ideal of having a single source of truth for a business is a rather difficult goal in a real-world business environment, it is a good design principle to minimize the amount of partially redundant data stores that may contradict each other. Data warehouses are meant to be used as the data source for multiple systems, so that all of them have access to the same data in the same format. [16, pp. 1-4]

The self-service business intelligence approach connects to the source systems directly from the business intelligence solution itself, performing the same data transformation steps to make the data usable for analysis as in the data warehousing approach, and storing the data set locally in the BI solution. This carries the benefit of connecting to only the data sources necessary for the given BI application, and in the absence of a data warehouse, is a faster method of getting started with BI development than building a data warehouse from the

ground up. However, the drawback is that this approach does not scale as well as data warehousing: as more BI development is performed under this approach, the data transformation steps must be performed for each BI application separately, leaving room for error. Users may perform different data transformation steps on the same data in different BI applications, leading to discrepancies in the analysis when compared to each other. Since the data is also stored locally in each BI application, this leads to duplication and redundancy, which can be an issue when storing the BI applications locally. [17] In terms of efficiency, data warehousing has clear advantages over the self-service BI approach, especially when using a local BI application such as Power BI Desktop and not the Power BI cloud service.

An alternative to data warehousing for big data is storing it in a data lake. A data lake is a collection of data from various systems in its raw form. Data lakes are highly compatible with the self-service BI approach, allowing analysts to make use of large masses of data that are not otherwise being taken advantage of. [18]

2.2 Extract, Transform, and Load: integrating data from source systems

In the previous section, two main approaches to building a data set for BI development were identified: consolidating the source data in a single data warehouse or connecting to the source data directly from the BI solution. The common thread between these two approaches is the need to connect to the source data and transform it into a form suitable for use in analysis. This is achieved through the Extract, Transform, and Load (ETL) process. [16, pp. 173-174] Generally considered a data warehousing term, the same steps can be identified in the self-service BI approach. In this case, the term ‘data integration’ can be used instead, but the end goal and steps taken to get there are similar. [17]

The basic principle behind the ETL process is connecting to the source data systems from an ETL system, then either staging the data for transformation or performing the transformations in memory. The transformed data is then loaded into the target system, the data warehouse. In a self-service BI approach, there is no separate ETL system, as the features are built in to the BI development tool itself, and the data is loaded into the BI model locally, but the same principles apply. [16, pp. 174-177]

One of the challenges of extracting data is avoiding stressing the source systems too much to minimize the amount of disruptions caused to normal day-to-day operations. The types of online transaction processing (OLTP) systems that tend to be used as data sources in BI solutions are typically not designed to transfer large amounts of data at once. Therefore, the ETL process should be designed to extract data incrementally: only updating the fields in the data set that have changed since the last update. The incremental batch updates should also be performed as infrequently as the business requirement of the target application allows, such as only once a day, as opposed to triggering an update on every new transaction. Finally, scheduling the data extraction step to take place during a period of low workload on the source system is a good design choice to keep the extraction times low. [16, pp. 173, 181-184]

Data transformation refers to changing the data from the source system during the ETL process to match the needs of the target system. The most typical form of data transformation is converting it into the desired data format, such as converting date data stored as string format to a date format. Other types of transformations include splitting and combining columns, trimming whitespace or leading zeros, and aggregating. For instance, a single date field might be split into a year, month, and day fields, or vice versa. [16, pp. 173-174] During the data transformation step, data cleaning may also be required. Data cleaning refers to removing invalid data from the result set, such as social security numbers that aren't formatted properly. This is necessary for maintaining the data quality of the BI solution. [19]

The final step of the ETL process is loading the data into the target system. This is a straightforward process: as long as the data transformations have been performed correctly and invalid data has been cleaned from the batch, it can be stored in the data warehouse or other target system with no errors. Ideally this is done with no delay, by extracting the data from the source system into the memory of the ETL system and performing the transformations there. However, temporarily staging the data in the file system or a database may be necessary. In the case of a self-service BI approach, the ETL system and the target system are the one and the same. [16, pp. 173-174] The entire ETL process as a concept can be seen visualized in figure 1, where the target system is a data warehouse serving multiple BI systems and the data sources are a customer relationship management (CRM) system, an ERP system, and a flat file directory on a file server.

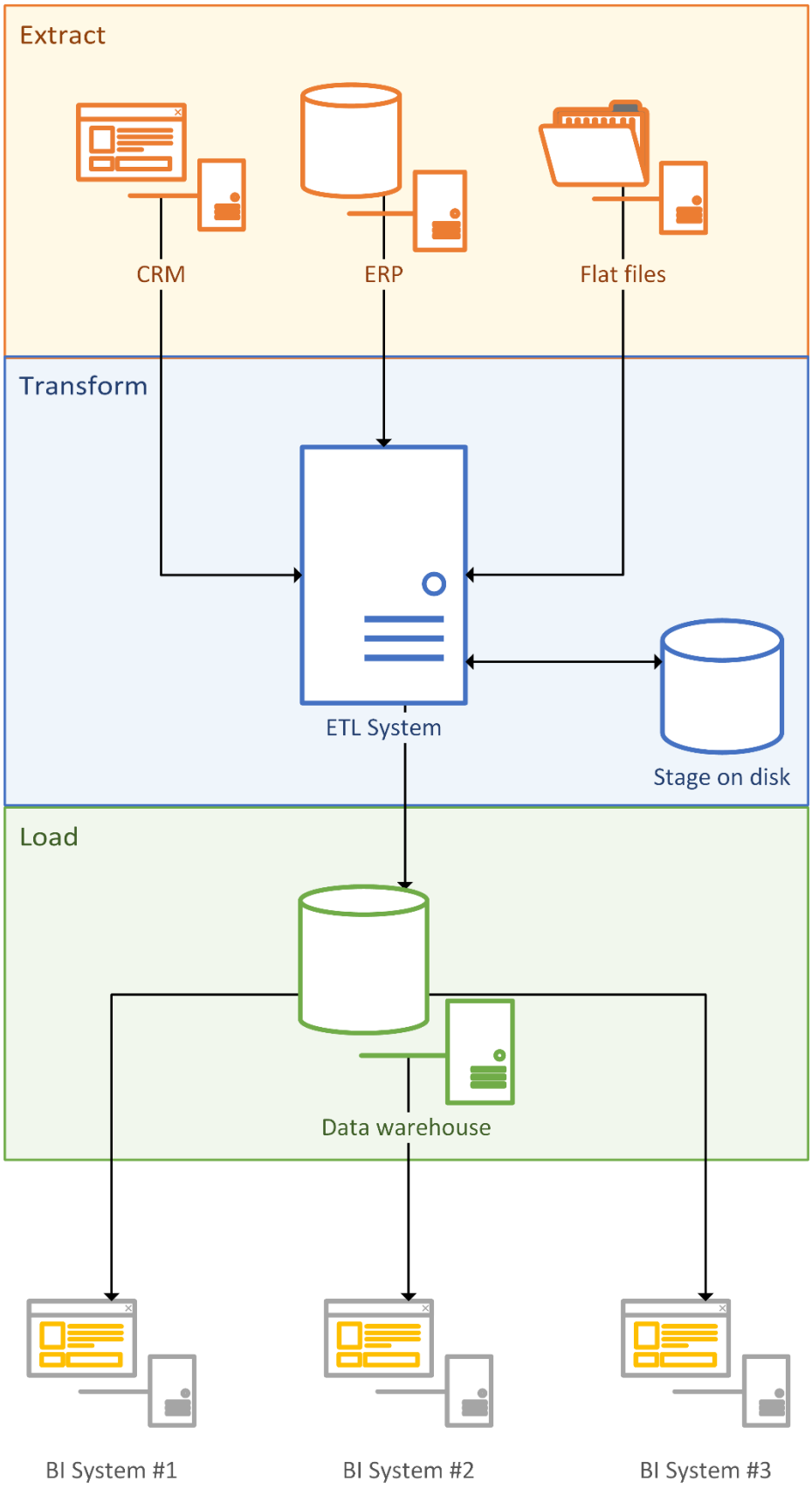


Figure 1. Extract, Transform, and Load (ETL) conceptual process diagram [16, p. 174]

Microsoft Power BI is suited for both accessing data stored in a data warehouse and building a self-service BI solution. Power BI Desktop provides easy connectivity to the most popular data sources used in BI development today through a data importing wizard with no scripting or programming knowledge required. Users may choose to connect live to a single data source with a 'DirectQuery' data model, such as when using a data warehouse, to avoid importing the data to the model and only query it from the data source on demand. The other option is to build an 'Import' data model, where users can connect to multiple data sources, and store the data in the model, with either regularly scheduled or manual updates. The full list of data connectors available in Power BI Desktop is listed in table 1. Most of the data connectors in the list require installing provider or driver software on the machine where data connectivity is required to use. The generic ODBC (Open Database Connectivity), OLE DB (Object Linking and Embedding, Database), Web, and Blank Query connectors can be used to connect to data sources that are not found in the list. [7, pp. 133-134]

Table 1. Data sources supported by Power BI Desktop as of June 2019 [20]

Data Source Type	Data Sources
File	Excel, CSV, XML, delimited or fixed-length text files, JSON, PDF files, list of filesystem or SharePoint folder files
Database	Microsoft SQL Server, Microsoft Access, SQL Server Analysis Services, Oracle, IBM Db2, IBM Informix, IBM Netezza, MySQL, PostgreSQL, Sybase, Teradata, SAP HANA, SAP Business Warehouse Application Server, SAP Business Warehouse Message Server, Amazon Redshift, Impala, Google BigQuery, Vertica, Snowflake, Essbase, Atscale cubes, BI Connector, Dremio, Exasol, Indexima, Intersystems IRIS, Jethro, Kyligence Enterprise, MarkLogic, other ODBC-compliant databases
Power BI Service	Power BI datasets, Power BI dataflows
Azure	SQL Database, SQL Data Warehouse, Analysis Services, Blob Storage, Table Storage, Cosmos DB, Data Lake Store, HDInsight (HDFS), HDInsight Spark, HDInsight Interactive Query, Data Explorer, Cost Management

Online services	SharePoint Online list, Microsoft Exchange Online, Dynamics 365 Online, Dynamics NAV, Dynamics 365 Business Central (online / on-premises), Common Data Service, Azure Consumption Insights, Azure DevOps, Azure DevOps Server, Salesforce (objects / reports), Google Analytics, Adobe Analytics, appFigures, Data.World, Facebook, GitHub, MailChimp, Marketo, Mixpanel, Planview Enterprise One – PRM / CTM, Planview Projectplace, Quickbooks Online, Smartsheet, SparkPost, Stripe, SweetIQ, Twilio, tyGraph, Webtrends, Zendesk, Emigo Data Source, Entersoft Business Suite, Industrial App Store, Intune Data Warehouse, Microsoft Graph Security, Quick Base, TeamDesk
Other	Web, SharePoint list, OData feed, Active Directory, Microsoft Exchange, Hadoop File (HDFS), Spark, R script, Python script, ODBC, OLE DB, BI360, Denodo, Information Grid, Paxata, QubolePresto, Roamler, SurveyMonkey, Tenforce, Workforce Dimensions, Blank Query

After extracting the data from the source systems using the data connectors listed in table 1, the next step is to perform the necessary data transformations using Power BI Desktop's Query Editor tool. The Query Editor allows users to perform the typical transformation and data cleaning steps expected of an ETL process, such as renaming columns, changing column data formats, removing unnecessary columns, splitting or combining columns, removing rows with errors or duplicates, combining rows, and trimming rows. [7, pp. 138-140] Users can also perform more advanced transformations using the Advanced Editor, where the necessary transformation steps can be written in the Power Query M formula language. [21] When all the necessary transformations are done and the user is satisfied with the preview of the results, the transformed data is loaded into the Power BI data set where it can be utilized, completing the ETL process [7, p. 140].

2.3 Data model design: key terminology and techniques

In this section, the basic terms and design choices related to data modeling are introduced. To perform analysis in the BI solution on data extracted from different source systems stored in several tables that are somehow related to each other, a data model is necessary. A well-

designed data model provides usability benefits for the designer of the BI solution and performance benefits for the end user. By following the design techniques and considerations explained in this section, common problems and challenges in data modeling can be avoided or mitigated.

2.3.1 Relationships

Upon completion of the ETL process, the transformed, ready-to-use, data is loaded into tables in the target system. This disconnected set of tables can be formed into a data model by creating relationships. A data model is simply a set of tables that are connected to each other with relationships by using keys: columns that are shared between each connected pair of tables. [22, pp. 24-25] A primary key uniquely identifies each row in the table, and a foreign key references a value that is found in another table. [7, p. 121]

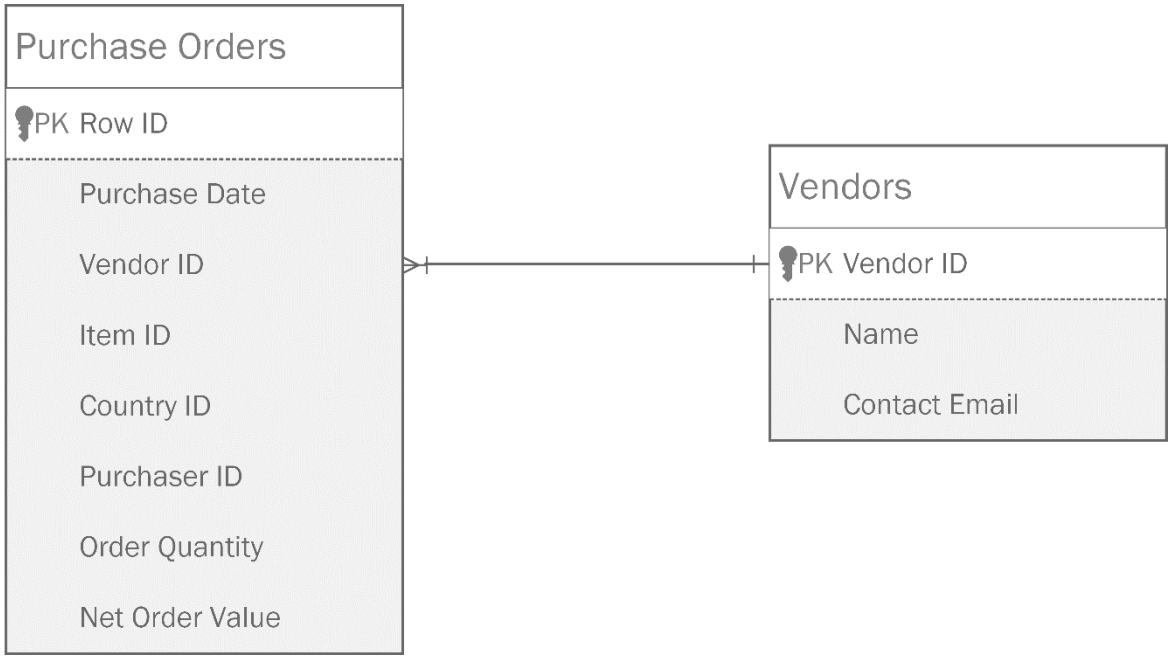


Figure 2. Sample data model formed by two tables linked by a many-to-one relationship on the ‘Vendor ID’ key

For example, take the sample data model presented in figure 2. There is a ‘Purchase Orders’ table that contains individual purchase order row level data, such as ‘Order Quantity’, ‘Net Order Value’, ‘Item ID’, and ‘Vendor ID’. The ‘Vendor ID’ column in the ‘Purchase Orders’ table is a foreign key that references the primary key of the ‘Vendors’ table: ‘Vendor ID’.

By creating a relationship between these two columns, data from the ‘Vendors’ table can be combined with the data from the ‘Purchase Orders’ table and vice versa. For instance, displaying the ‘Name’ field from the ‘Vendors’ table alongside ‘Purchase Orders’ data, or summarizing total ‘Net Order Value’ from the ‘Purchase Orders’ table alongside a list of vendors from the ‘Vendors’ table. [22, pp. 27-28]

Relationships in a data model have a direction and a cardinality. To find out the direction of a relationship, it is necessary to define which table is the source and which table is the target of the relationship. The source of the relationship is the table where the value to be looked up in the other table is taken from, and the target is the table where the search is performed. In the sample data model portrayed in figure 2, ‘Purchase Orders’ is the source of the relationship between it and the ‘Vendors’ table, which is the target of the relationship. This indicates that the ‘Vendor ID’ value is taken from the ‘Purchase Orders’ table, and the attributes of the vendor are looked up in the ‘Vendors’ table. [22, pp. 27-28]

The cardinality of the relationship refers to how many times each value of the key column can appear on either side of the relationship: their uniqueness. There are three types of cardinality, barring optional relationships and not accounting for the direction of the relationship: one-to-one, one-to-many, and many-to-many. Optional relationships allow for no occurrences on either end of the relationship, for example one-to-zero-or-one. Again, returning to the sample data model in figure 2, it can be determined that the cardinality between the ‘Purchase Orders’ and ‘Vendors’ tables is many-to-one: each purchase order row can only reference one vendor, but each vendor can have multiple purchase orders. Thus, the ‘Vendor ID’ column can have duplicates in the ‘Purchase Orders’ table but it must be unique in the ‘Vendors’ table. This is denoted in the model diagram with a forked line and a dash next to the ‘Vendor ID’ column in the ‘Purchase Orders’ table, meaning ‘one or many’, and a dash on the line next to the ‘Vendor ID’ column in the ‘Vendors’ table, meaning ‘one’. [23, pp. 75-78]

2.3.2 Star schema

However, very simple data sets such as the one depicted in figure 2 are not common in real-world business intelligence solutions. To meet business needs, there would be many more

informational assets to track, such as items, purchasers, and countries. As more and more tables are added to the data model and it becomes more complex, the value of utilizing a data-modeling technique called a star schema becomes apparent. In the context of data modeling, the term schema refers to the kind of data each table contains and how the relationships between them are formed. [7, p. 118] When designing a data model according to the star schema, there are two categories that each table, or entity, in the data model should fall under: dimension tables and fact tables. Dimensions contain data with attributes; they are informational assets, such as vendors or items. Fact tables contain metrics about events that are related to dimensions, such as purchase orders. A sample star schema data model can be seen in figure 3. In it, the 'Vendors', 'Items', 'Purchasers', 'Country', and 'Calendar' tables are dimensions, and their columns are attributes. 'Purchase Orders' is a fact table. Each purchase order row is an event involving each of the dimensions, and the metrics related to each event are 'Order Quantity' and 'Net Order Value'. [22, pp. 35-36]

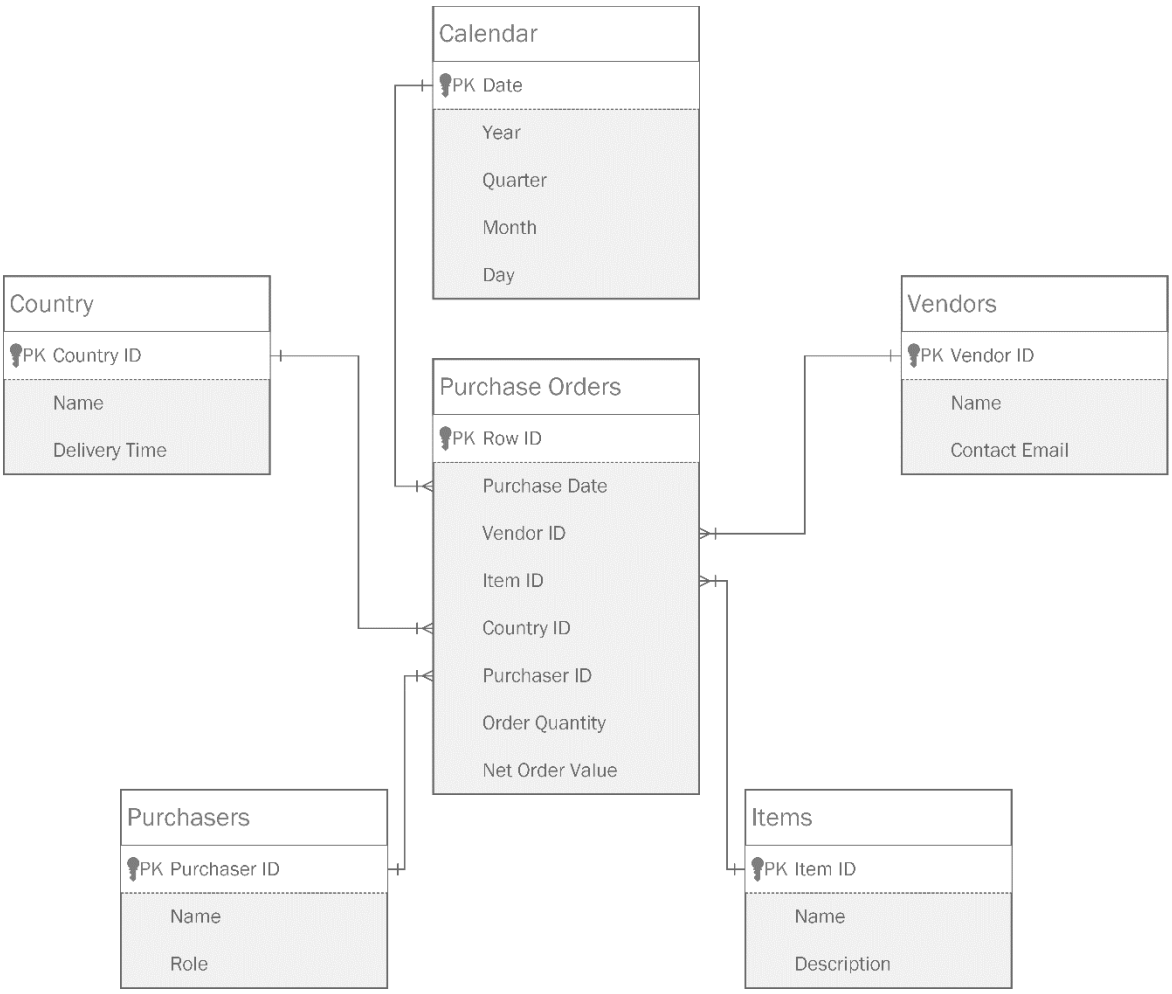


Figure 3. Sample star schema data model. [7, p. 118]

The star schema is named after the shape the data model forms when you place the fact table in the middle and its dimensions around it, as can be seen in figure 3. In a star schema, dimensions can only be related to fact tables, and not to other dimensions. By conforming to the star schema, it is possible to prevent ambiguity in the data model relationships while minimizing the amount of unnecessary repetitions of the same value in a column on different rows, making it a very good choice from an efficiency standpoint. [22, pp. 37-38]

2.3.3 Granularity

An important variable to consider when designing a data model is granularity. Granularity refers to the level of detail in each table, or data set. Increasing the level of granularity reduces the level of detail in the data, and vice versa. For a simple example of how granularity levels work and how they affect the size of the data set, take a date table containing data for a single non-leap year: the lowest level of granularity would be the individual day level, resulting in a data set with 365 rows. Increasing the level of granularity results in less specific data and fewer options for analysis but reduces the size of the data set: going up to week number level granularity reduces the number of rows in the data set to 52, month level granularity to 12, quarter level granularity to 4, and the year level granularity to just 1. [22, pp. 20-24]

In the data model shown in figure 3, the ‘Purchase Orders’ fact table is at the individual purchase order item level, which is the lowest level of granularity available in this scenario. Because a single purchase order typically has multiple items, increasing the level of granularity in this case would mean going up to purchase order level by removing the ‘Item ID’ field from the table, greatly reducing the size of the data set at the cost of losing access to the information of which items were included in each purchase order. Choosing the right level of granularity is the key to fulfilling the business requirements of the data model and balancing the trade-off between the available level of detail and the resulting performance costs. [22, pp. 20-24]

2.3.4 Data normalization and denormalization

Data normalization is a design technique that assists in the creation of star schema data models and optimizing performance. Separating attributes from fact tables into dimensions

is a key element of building a star schema data model, and that is the result of normalization. For example, in the star schema data model presented in figure 3, the ‘Name’ and ‘Contact Email’ information of the vendor fulfilling the purchase order have been removed from the ‘Purchase Orders’ fact table and are instead found in the ‘Vendors’ dimension table through the relationship formed between the ‘Vendor ID’ fields of each table; in other words, these columns have been normalized. The benefit of this is that the ‘Name’ and ‘Contact Email’ fields are stored in a single row for each vendor despite being connected to multiple purchase order rows, avoiding unnecessary repetitions of the same information and reducing the size of the data set on disk. This also means that if the contact email information of a vendor were to change, it would only have to be updated in one place, and all the purchase orders related to the vendor would automatically display the updated information through the relationship. [22, pp. 32-35]

Denormalization is the reverse of normalization: storing the attributes in the fact table instead of separating them into a dimension. In the previous example, this would mean removing the ‘Vendors’ dimension table entirely and adding the ‘Contact Email’ and ‘Name’ fields into the ‘Purchase Orders’ fact table. This can be necessary to prevent ambiguity in the data model and is helpful to the designer working with the data model to develop a business intelligence solution, since all the required fields for analysis are found in one place, instead of having to look them up from multiple tables. There is no single answer for finding the right level of denormalization but considering the number of attributes related to each entity and the data format of each attribute is a good way to find the right balance between performance and ease of use. [22, pp. 32-35]

2.3.5 Filter propagation and ambiguity

To understand data model ambiguity, one must first grasp how filters propagate through the data model. This section is specific to Power BI because it uses the DAX language to perform calculations and ambiguity is not allowed in a data model that uses DAX. For an example of how filter propagation works, consider the following data model: two completely denormalized fact tables, ‘Purchase Orders’ and ‘Reclamations’, as shown in figure 4. [22, pp. 71-72] The data in the example figures used in this section is all mockup data generated using Mockaroo [24], a web service that enables realistic random data generation.

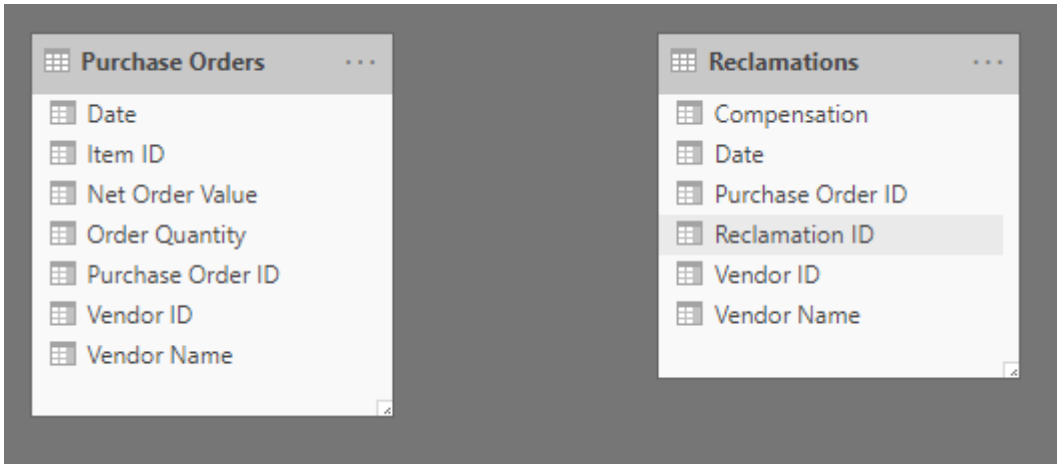


Figure 4. Sample denormalized data model consisting of two fact tables. [22, p. 61]

This completely denormalized data model is fine if purchase orders and reclamations are to be analyzed separately, but if users want to view the purchase order volume and reclamations of a given vendor side by side in a single visualization, the two tables must be linked somehow, or Power BI will fail to filter the data correctly, as shown in figure 5. [22, p. 61]

Purchase Orders

Vendor Name	Purchase Orders	Net Order Value
Vendor 14	49	7 151 801 €
Vendor 18	49	7 087 189 €
Vendor 20	47	6 850 535 €
Vendor 15	45	6 467 584 €
Vendor 19	47	6 328 605 €
Vendor 7	42	6 227 959 €
Total	**943**	**124 540 568 €**

Reclamations

Vendor Name	Reclamations	Compensation
Vendor 3	63	314 940 €
Vendor 6	51	286 623 €
Vendor 16	46	248 853 €
Vendor 2	48	245 830 €
Vendor 9	48	235 405 €
Vendor 10	44	228 379 €
Total	**1000**	**5 056 889 €**

Vendor Fact Sheet

Vendor Name	Purchase Orders	Net Order Value	Reclamations	Compensation
Vendor 14	49	7 151 801 €	1000	5 056 889 €
Vendor 18	49	7 087 189 €	1000	5 056 889 €
Vendor 20	47	6 850 535 €	1000	5 056 889 €
Vendor 15	45	6 467 584 €	1000	5 056 889 €
Vendor 19	47	6 328 605 €	1000	5 056 889 €
Vendor 7	42	6 227 959 €	1000	5 056 889 €
Total	**943**	**124 540 568 €**	**1000**	**5 056 889 €**

Figure 5. Data only filters when viewed separately in denormalized model. [22, p. 62]

The data is filtered correctly in the separate ‘Purchase Orders’ and ‘Reclamations’ table visualizations, but the reclamations data is all wrong in the ‘Vendor Fact Sheet’ table visualization, where data from both tables is displayed side by side. The cause of this problem is that the ‘Vendor Name’ field has been brought from the ‘Purchase Orders’ table in the combined table visualization, and since there is no relationship between the two tables, Power BI doesn’t filter the ‘Reclamations’ and ‘Compensation’ columns at all, displaying the total value for all vendors on each row instead. [22, p. 62]

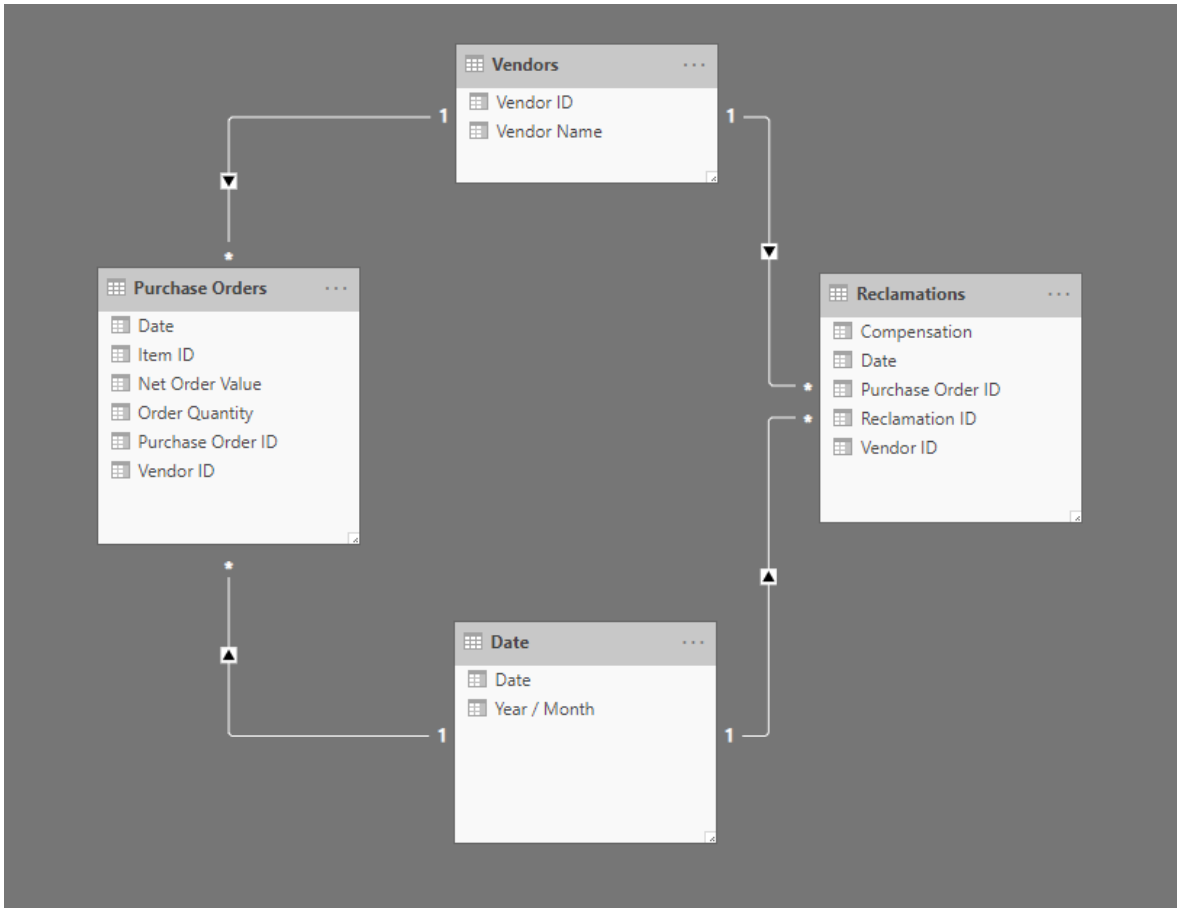


Figure 6. Normalized version of the data model from figure 5. [22, p. 68]

Solving this problem requires changing the data model by normalizing the fact tables. This results in the creation of two new dimensions: ‘Vendors’ and ‘Date’. The two fact tables are linked through these new dimensions using the ‘Vendor ID’ and ‘Date’ columns as keys for the relationship, as shown in figure 6. As a result, the data now filters correctly in the ‘Vendor Fact Sheet’ table, as shown in figure 7, since the ‘Vendor Name’ information is taken from the ‘Vendors’ dimension that has a relationship with both fact tables. [22, p. 68]

Vendor Fact Sheet

Vendor Name	Purchase Orders	Net Order Value	Reclamations	Compensation
Vendor 14	49	7 151 801 €	40	208 729 €
Vendor 18	49	7 087 189 €	35	174 662 €
Vendor 20	47	6 850 535 €	36	190 053 €
Vendor 15	45	6 467 584 €	39	196 184 €
Vendor 19	47	6 328 605 €	40	189 339 €
Vendor 7	42	6 227 959 €	40	195 166 €
Total	**943**	**124 540 568 €**	**1000**	**5 056 889 €**

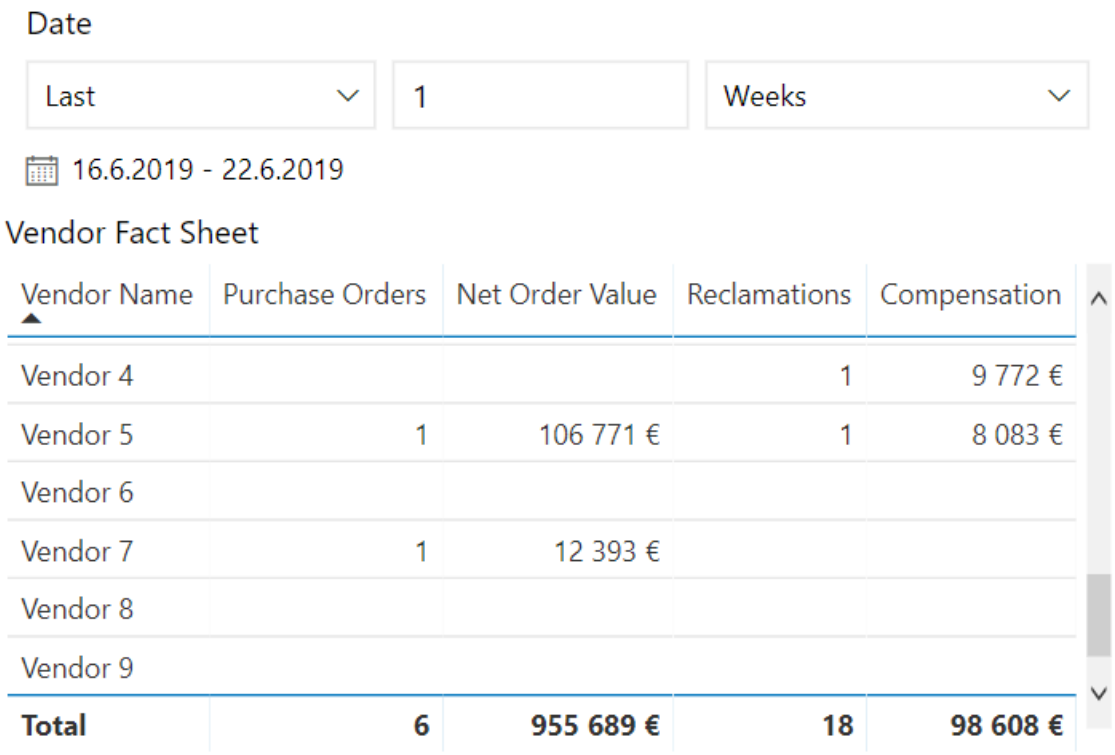
Figure 7. Data filters correctly after normalization [22, p. 69]

The way filter propagation works in this example is that filtering the ‘Vendors’ table applies the same filter to the ‘Purchase Orders’ and ‘Reclamations’ tables, but not the other way around. However, this can be changed with the relationship’s cross-filter direction setting in Power BI. The default setting for cross-filter direction is single, and with this setting filter propagation only occurs from the destination table to the source table of the relationship. The other option is to use the bidirectional cross-filter direction, which allows filter propagation to also occur from the source table to the destination table. [22, pp. 69-71]

The benefit of enabling bidirectional cross-filtering is that now any other filters affecting the source table are also reflected in the destination table. In this example, where each relationship’s cross-filter direction is set to single, filtering the ‘Date’ dimension propagates the filter to the ‘Purchase Orders’ and ‘Reclamations’ tables, but not the ‘Vendors’ table. Therefore, without bidirectional filtering, the ‘Vendors’ table will be completely unfiltered even if a date filter is applied, as shown in figure 8. [22, pp. 69-71]

To allow filter propagation from the ‘Date’ table to the ‘Vendors’ table, the cross-filter direction for the relationship between the ‘Vendors’ table and either the ‘Purchase Orders’ or ‘Reclamations’ table must be set to bidirectional. However, changing the cross-filter direction runs the risk of adding ambiguity to the data model. A data model that is ambiguous has multiple paths connecting any two entities in the model. Power BI only allows one active

filtering path between tables in a data model, because the DAX language has several functions that work on the relationships in the data model, and these functions work without specifying which route to use. Therefore, ambiguity is to be avoided when working with Power BI. This example data model would become ambiguous if the cross-filter direction of the relationship between both the 'Purchase Orders' and 'Reclamations' tables and the 'Vendors' table was changed to bidirectional, because this would create two paths for filtering the 'Vendors' table from the 'Date' table. There are ways to get around this limitation in calculations by using inactive relationships and explicitly referring to them in the DAX code, but unless it is necessary for a particular calculation, it is better to remove inactive relationships from the data model. [22, pp. 71-73]



Vendor Name	Purchase Orders	Net Order Value	Reclamations	Compensation
Vendor 4			1	9 772 €
Vendor 5	1	106 771 €	1	8 083 €
Vendor 6				
Vendor 7	1	12 393 €		
Vendor 8				
Vendor 9				
Total	**6**	**955 689 €**	**18**	**98 608 €**

Figure 8. Filtering the date table does not propagate to the vendors table unless cross-filter direction is changed to bidirectional in the data model to create a filtering path.

2.4 Data Analysis Expressions: expanding the data model

Business intelligence analysis often requires performing additional calculations on the data model. The source data might not include all the fields required to implement the data visualizations and KPIs that analysts need access to, but rather the fields that the necessary values can be calculated with. For instance, take supplier on-time performance: if an analyst needs to know the percentage of how many purchase orders arrived on time out of all purchase orders for a given supplier or time frame, and the base data model only provides access to the dates when a purchase order arrived and when the agreed delivery date was, it is necessary to expand the data model to calculate this new value based on these two fields. Power BI's solution to expanding the data model is DAX, a language inherited from SQL Server Analysis Services. [25, p. 1]

Data analysis expressions, or DAX, is a programming language that runs on top of the VertiPaq database engine, an in-memory database that stores the data model in Power BI. [25, p. 399] DAX is designed to perform business formula calculation on a data model. [25, p. 1] It bears resemblance to the Excel formula language, making it more accessible to business users; being a functional language, all expressions in DAX are function calls, and there are no statements, loops, or jumps, which are common features in several programming languages. [25, pp. 5-8] There is a large number of functions available for business intelligence analysis in DAX: time-intelligence functions, filter functions, logical functions, and statistical functions, to name a few categories. [26]

In DAX, there are two important concepts to understand when planning to expand a data model by defining new calculations: calculated columns and measures. Adding a calculated column to a table in a data model creates a new column, functionally identical to any other column in the table, except that it is based on a DAX expression, and not loaded along with the source data. Calculated columns can be used to display values, filter the table, and create relationships in the data model, just like any other column. The evaluation context of a calculated column's DAX expression is the current row, meaning that it is possible to directly refer to the values of other columns on the same row in the table when writing the expression, but not the values of different rows. Calculated columns are static: they are only computed when the data stored in the model is refreshed, making them a good choice for

complex calculations that require more compute time, but ill-suited for values that need to be re-calculated when the context changes, such as when time and date filtering is applied, because filtering does not trigger a data refresh. Measures, on the other hand, are used when aggregating values from several rows in the table and are calculated dynamically to match the current filter context. The evaluation context of a measure is the context of the cell, which depends on the particular DAX expression. Directly referring to any column's values in a measure is not possible, and functions must instead be used to calculate values. Measures can be used to display values, but unlike calculated columns, they cannot be used to filter the table or create relationships, because they are not connected to any row. [25, pp. 22-24]

To summarize, calculated columns should be used if the new value is row-specific or is required to be used in filtering the table or creating new relationships but doesn't need to be re-computed when changing the filter context. Measures should be used when calculating aggregate values such as percentages of a selection that need to be updated dynamically along with filter context changes. [25, p. 25] Returning to the supplier on-time performance example from earlier, one way to implement it in DAX would be to create a measure that counts the number of rows where the purchase order's date of arrival was on or before the agreed delivery date and dividing that number with the total number of rows in the table. A measure is the better option for this scenario, because the on-time performance is aggregate data from multiple rows, and it will change based on filtering by date or supplier.

2.5 Measuring performance and methods for optimization

For a business intelligence solution that is intended for daily operative use, minimizing the disruptions to work flow is important. Technical performance issues direct the users' attention to the technology in a negative manner, distracting them from the tasks that they are trying to accomplish. To determine how a system's user-perceived performance could affect user satisfaction, the system response time (SRT) metric can be used. [27]

SRT is the amount of time that passes between the user submitting a request to an interactive system and receiving the result. The maximum acceptable value for a given task's SRT is related to the perceived complexity of the task: it is acceptable for more complex tasks to take longer to complete without negatively affecting user satisfaction. Doherty and Sorenson

proposed a framework for categorizing system response times and how likely it is for the user to disengage from the task flow in each case, as seen in table 2. [27]

Table 2. SRT framework proposed by Doherty and Sorenson [27]

Attention	Category Name	SRT Range	Description
Attentive	Instantaneous	< 300 ms	User feels like they're in direct control.
	Immediate	300 ms – 1 sec	Perceived as easy to perform.
	Transient	1 sec – 5 sec	Perceived as requiring simple processing, but user feels that they are making progress if given feedback. Unlikely to disengage from task flow.
	Attention Span	5 sec – 10 sec	Perceived as requiring some complex processing, but user can remain engaged to task flow if given informative feedback.
Non-Attentive	Non-attentive	10 sec – 5 min	Perceived as requiring complex processing. Users likely to disengage and multi-task.
	Walk-away	> 5 min	Perceived as requiring intensive processing. Users would not stay engaged to the task flow.

For a BI solution that is used daily by operative personnel to assist in their tasks, maintaining work flow is important. Therefore, the SRT of the BI solution should be kept to the attentive categories in the SRT framework shown in table 2. Ideally, given the powerful analytics engines that power the data models of today's BI solutions, the SRT should be kept to under 5 seconds to ensure no interruptions to work flow.

To measure system response time in Power BI, Microsoft added a performance analyzer tool to Power BI Desktop in the May 2019 feature update. It allows users to see how long it takes to load each visualization on a page after any action, such as opening the page or

filtering it, and which factors contributed to the length of the load time. The performance analyzer tool makes it possible to identify the performance bottlenecks in a report, such as poorly optimized DAX measures or underlying issues in the data model. [28]

In addition to system response times, the memory usage and file size of the Power BI report are important metrics for gauging efficiency, and they can be easily measured. Since the VertiPaq database engine that Power BI uses loads the data set in memory, a large and poorly optimized data model will take up considerable amounts of memory [25, p. 399]. For a report that is distributed locally using Power BI Desktop and not using the Power BI cloud service, it is important to keep the memory usage and file size under control to ensure equal performance for all end users, since their hardware specifications will vary.

There are a number of ways to optimize performance in Power BI. The simplest method is to limit the amount of data that is imported into the model or displayed in a visualization. Limiting the number of visualizations and slicers on a report page will improve SRT. Removing unused tables or columns from the data model is effective for lowering the memory usage and file size of the report. Avoiding including columns with high cardinality in the data model and not performing distinct counts on said fields is a good practice. The term cardinality in the context of a data column refers to how many unique values the column contains: high cardinality means that there are few duplicates in the column's rows, and most values are unique. Cardinality can be lowered by removing unnecessary precision by splitting fields and rounding or truncating values. For example, a datetime field could be split into year, month, day, and time fields. A column's data format choice can also affect performance: integer fields should be used instead of strings if possible, such as for ID fields. Special care should be taken when working with iterative DAX functions that test every row in a table, such as RANKX, because their performance costs can increase exponentially as the size of the data set grows over time. [29]

Finally, the degree of normalization in the data model should be considered. Relationships in a data model come with a performance cost, because the database engine must propagate the filter between each table involved in the relationship. With bidirectional cross-filtering, the number of tables being filtered can be quite high. Therefore, unnecessary related tables should be denormalized where possible in the data model. [25, p. 434]

3 IMPLEMENTATION

This chapter details the background of the thesis project and the necessity of developing new tools for supply chain management at ANDRITZ Savonlinna Works Oy, the development process of the business intelligence solution that would meet the needs of the company, and the end results of the development process.

3.1 Background: the need for new tools to meet manufacturing demands

ANDRITZ Savonlinna Works Oy is a workshop operating in the pulp and paper industry that manufactures machinery and equipment for various pulp screening, cleaning, and bleaching processes, such as DD-Washers, drum filters, disc filters, cooking equipment, and screens, with the delivery scope ranging from single spare parts to large fiber line projects. The company also offer a variety of services, such as maintenance, modernizations, upgrades, and installations. As a part of the ANDRITZ Group, the company participates in large scale pulp and paper plant manufacturing projects all over the world. [30] Having been in operation since 1917, there is a wealth of experience and mastery at the company's disposal [31]. As the company moves toward a lean manufacturing process and planning its manufacturing process steps according to Takt time, it is important for the procurement department to deliver the necessary materials and components for manufacturing on time.

To do this, acquiring better tools for supply chain management was necessary. The procurement department was reliant on Excel models that required manual upkeep every week, taking up valuable time. As the Excel models had initially been intended to be a temporary solution and had overstayed their welcome by several years, they had grown quite bloated and slow. According to an estimate, maintaining the Excel models took approximately 400 - 450 hours every year. In addition, the Excel models only provided tools for retrospective analysis such as measuring on-time performance, and for continuous development of the procurement process, more proactive risk analysis tools were required.

As none of the pre-existing tools available to procurement personnel at ANDRITZ Savonlinna Works Oy were able to fulfill this growing demand, there was a need to develop an entirely new tool. To this end, a business intelligence development project was launched in January 2019 as the subject of this master's thesis. The first month was spent on-site in

Savonlinna, to familiarize the company and the procurement processes. To better understand the bottlenecks of the procurement process and which key performance indicators to implement in the new tool, the work started by modeling the procurement processes. When an understanding of the problem at hand was reached, the development could commence.

3.2 The development process of the BI solution

From the very beginning, the thesis project had a clear goal: to develop a BI solution providing interactive dashboards for each process of the procurement department at ANDRITZ Savonlinna Works Oy to assist in day-to-day supply chain management tasks and remove the need to maintain the old Excel models. The research questions and literature review of this thesis were based on reaching this goal in a way that was proven to be effective in research. The procurement department's activities can be classified under five processes: management, strategic procurement, purchasing planning, operative purchasing, and internal logistics. Each of these processes has its own key performance indicators and things to keep track of. Designing the requirements specification for a BI solution that meets each process's needs would require expert knowledge, and there also needed to be plenty of time to implement the solution. Therefore, the following division of labor was decided for the thesis: the company would provide the specifications on which key performance indicator visualizations and analysis tools would need to be implemented and the formulae on how to calculate the necessary values, and the thesis itself would focus on the technical implementation and how to design and optimize it.

The prerequisite for beginning development was gaining the approval of the IT department and ANDRITZ Group procurement. After presenting the business requirements that the BI solution would meet and determining that there were no other pre-existing tools that could answer them, beginning development with Microsoft Power BI Desktop was approved. However, there was a caveat: the development process was accepted to continue as a proof of concept, and not a fully supported implementation. In effect, this meant that there would be no direct access to the data warehouse containing the SAP ERP data required for most of the dashboards, and the method of distributing the BI solution to the end users would have to be Power BI Desktop, ruling out the Power BI cloud service. The data was still accessible by exporting it from SAP's graphical user interface (GUI), but it presented a considerable challenge for performing the extraction automatically. It could still be done by scripting, but

it introduced plenty of potential for errors. Once the thesis work concluded and the proof of concept BI solution was finished, the result would be aligned with pre-existing BI tools, and its functionality would either be implemented in said tools or approval would be granted to officially implement the proof of concept as its own entity, and this workaround would no longer be required. Until then, it was decided that some degree of manual upkeep would have to be accepted in the proof of concept phase. Regardless, the most important hurdles were cleared: gaining approval, getting the development environment set up, and having access to the data, even if it was in a roundabout manner. The development process could begin.

The development project involved five people, and it was organized in three tiers: the developer, the management, and the supervisors. The management team was formed of two people, and actively assisted in guiding the development on a daily basis by producing dashboard specifications and giving feedback. The supervising team was likewise formed of two people, and their role was to follow the long-term development of the project and give direction and feedback from the perspective of someone who doesn't see the development version as often as the developer and the management. Weekly meetings were arranged between the developer and the management, mostly in the form of Skype meetings, and face to face meetings with the developer, management and supervisors present were arranged once a month. During these meetings, the results of the development since the previous meeting would be discussed, the dashboards would be updated based on this feedback, and new development tasks were assigned from the backlog to be implemented during the next week. The development process itself was iterative and incremental in nature, and it was managed in week-long iterations; it was not unlike Scrum, the popular agile software development framework. In the Scrum model, development is performed in sprints that are iteration cycles that last up to a month, during which the requirements of the project cannot be changed. The functionality to develop during a sprint is selected from the backlog in the beginning of each sprint. There are three key roles in the Scrum model: the development team, the product owner, and the Scrum Master. The product owner represents the interests of the stakeholders, and the Scrum Master acts in the role of the project manager. [32]

The speed of development was greatly accelerated towards the end of the project, when the weekly Skype meetings between the management and the developer were replaced with face

to face development meetings. This rapid loop of developing dashboard prototypes with the product owner present and instantly receiving feedback and advice on the intricacies of the data being extracted from SAP proved highly fruitful. Some lone development time was required in between these development meetings to tackle more technical problems such as particularly problematic DAX measures to get the most out of the meetings. Dashboards that are used to perform interactive analysis of the data to identify risks in the were created during these development meetings, since the product owner could see the interactivity in action and explore the data, drawing conclusions about how users might want to perform analysis.

3.3 Results of the development process: dashboard showcase

The result of the development process was a Microsoft Power BI Desktop report file made using a self-service BI approach comprising 30 pages, including 20 dashboards, 6 navigation pages, 3 drill-through pages, and a diagnostic page displaying when the data in each table of the data model was last updated. The full list of dashboards and their intended use cases can be seen in table 3 at the end of this chapter. Some of these dashboards provide interactive tools for aiding in operative tasks and conducting risk analysis, while others, such as the Procurement overview dashboard in figure 9, offer a more traditional, static KPI-heavy view.

Figure 9. Procurement overview dashboard

In the beginning stages of the design process, the intent was to restrict the total number of dashboards to 10, limiting each process to a maximum of two dashboards. However, as the development work progressed, the sheer amount of KPIs to visualize proved to be a formidable design challenge when working with the 10 dashboards limit. The readability and usability of the dashboards suffered because of attempting to fit too much information on one page, and interactive dashboards in particular proved to require a considerable amount of space. As a result, the limit on the number of dashboards was lifted to give more freedom of design.

This doubled the number of dashboards, necessitating the implementation of a more refined navigation system. Power BI organizes report pages in tabs, much like Microsoft Excel, and in report files with fewer pages, it is a perfectly adequate navigation system. However, when the report file has many pages and not all the tabs can be displayed at once, users need to scroll the tab bar to find the page they are looking for, which can be a frustrating task. This problem was solved by using Power BI Desktop's bookmarks feature, which allows adding buttons to dashboards that direct users to a bookmarked page. Figure 10 depicts the front page of the BI solution, a navigation table of contents that has buttons that lead to a similar landing page for each process, as shown in figure 11.

Figure 10. Navigation front page containing buttons that lead to bookmarked pages

Figure 11. Operative purchasing navigation page

Operative purchasing personnel have a greater need for tools that assist in their daily work than KPI dashboards that only tell how things have gone in the past. All dashboards under the Operative purchasing section are analysis tools, such as the dashboard in figure 12, which assists purchasers in selecting which purchase requisitions to prioritize from the queue.

Figure 12. Open purchase requisitions queue analysis dashboard

The open purchase requisitions queue analysis dashboard is a scatter plot, where the circles represent open purchase requisitions, the X-axis value is the time they have spent in the queue in days, and the Y-axis represents the remaining time to deliver they have. The size of the circle indicates how many items are included in the purchase requisition: a larger circle meaning more purchase requisition items. The colors have no significance other than to help separate closely clustered purchase requisitions from one another. There are several guidelines in the scatter plot to assist in picking out the purchase requisitions that are most in need of being purchased to prevent delays. The dashed lines on the Y-axis from top to bottom are 'Less than 6 months remaining time to deliver', 'Less than 3 months remaining time to deliver', 'Less than a month remaining time to deliver', and 'Less than a week remaining time to deliver'. This refers to how much remaining time to deliver the circles below each line have. Any circles underneath the solid red line are already late from their planned delivery date. Any circles found in the red area of the scatter plot have been longer in queue than they have remaining time to deliver. These two values are not directly related, but the shaded area helps guide the user's gaze towards the purchase requisitions that most need to be removed from the queue.

If an operative purchaser wanted to find out more about a given purchase requisition - or circle - on the open purchase requisitions scatter plot in figure 12, all they need to do is right click the circle, select 'Drillthrough', and click 'Purchase requisition details', as shown in figure 13. This directs the user to a drill-through page depicted in figure 14 displaying detailed data about the selected purchase requisition in tabular form. The blue arrow in the top left corner of the drill-through page returns the user to the open purchase requisitions dashboard, making for a quick and convenient way to look up details while maximizing the space available for the scatter plot visualization.

The controls at the top of the page in figure 12 underneath the company logo bar are called slicers, and they filter the data that is displayed on the dashboard. In this case, they would filter the data points displayed on the scatter plot by limiting the purchasing group to one or more of the available selections, or by setting upper or lower limits for the axis values, time in queue and remaining time to deliver.

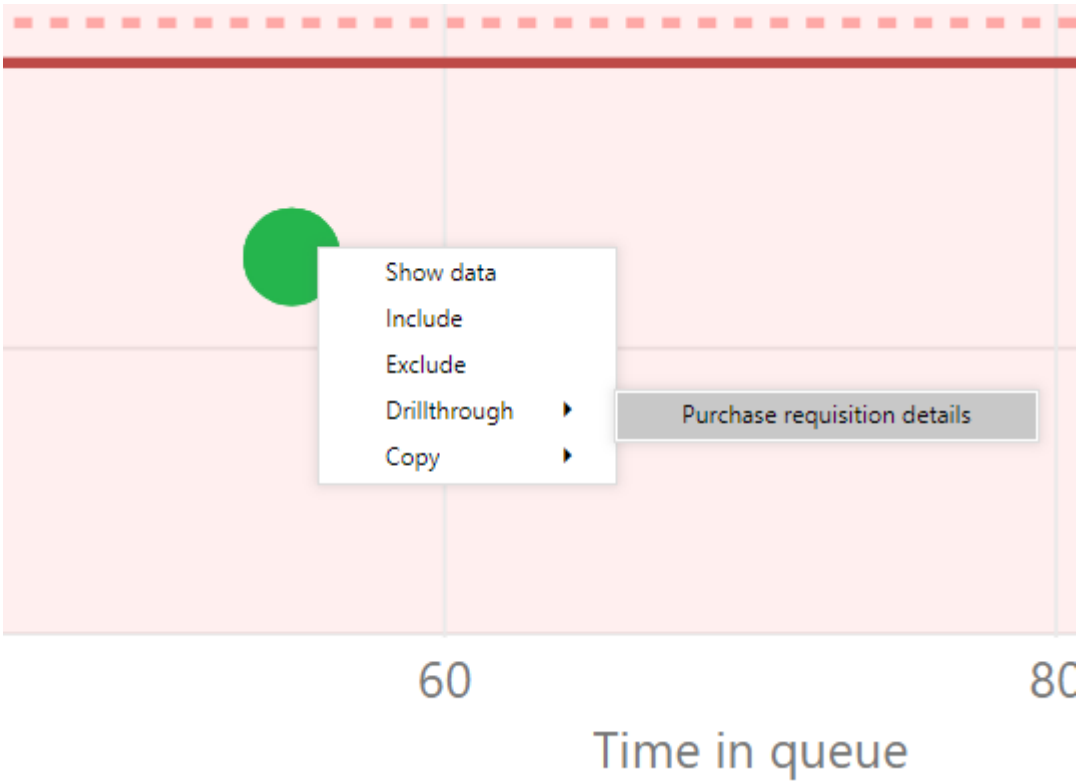


Figure 13. Getting to the details of a purchase requisition by using drill-through

COMPANY LOGO

Purchase Requisition	Purchasing Group	Andritz WBS Element	Project Code	Remaining time to deliver	Time in queue	Delivery date	Material Description	Short Text	Release date
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	PLATE L 888 W 942 MM		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	PLATE L 2016 W 2016 H 40 MM		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	PLATE L 1187.5 W 533 MM #2#		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	HALF COUPLING ASMEB16.11 3000 NPS=3/...		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	PLUG SQUARE ASMEB16.11 3000 NPS=1/2"...		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	TUBE_PIPE L 140 MM #2#		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	HALF COUPLING ASMEB16.11 3000 NPS=1/...		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	FLAT BAR L 248 W 50 H 6 T 6 MM		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	ROUND BAR L 766 D 16 MM #2#		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	SHEET L 1119 W 157 MM #2#		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	PLATE L 150 W 100 H 95 MM		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	PLATE L 1524 W 914 MM #2#		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	PLATE L 525 W 559 MM #2#		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	PLATE L 2142 W 1128.5 MM #2#		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	REINFORCEMENT PLATE L 944 W 944 H 35...		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	SHEET L 546 W 95 T 3 MM #2#		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	TUBE_PIPE L 685 D 21.3 MM #2#		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	PLUG SQUARE ASMEB16.11 3000 NPS=1/2"...		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	THREADED COUPLING L 48 W 40 H 40 MM #2#		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	PLATE L 203 W 154 MM #2#		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	PLATE L 1025 W 947 MM		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	SHEET L 1522 W 147 T 3 MM #2#		25.4.2019
[redacted]	FIL	[redacted]	[redacted]	-34	55	16.5.2019	PLATE L 219 W 25 T 25 MM #2#		25.4.2019
Total									

Figure 14. Purchase requisition details drill-through page

Another example of an interactive analysis tool is the open purchase orders analysis dashboard, shown in figure 15. This tool allows operative purchasers to perform risk analysis on upcoming projects to expedite open purchase orders that are at risk of being delayed and try to find causes for open purchase orders that are delayed. The X-axis on the column chart is time in 'Year / month' format, and the Y-axis shows how many open purchase orders there are for each month. The table below the column chart displays data about each open purchase order item that is currently visible in the bar chart. The dashboard provides multiple slicers for filtering the data, including the purchasing group, whether the supplier has confirmed the delivery date or not, the project the purchase order is for, and the purchaser who originally made the purchase order.

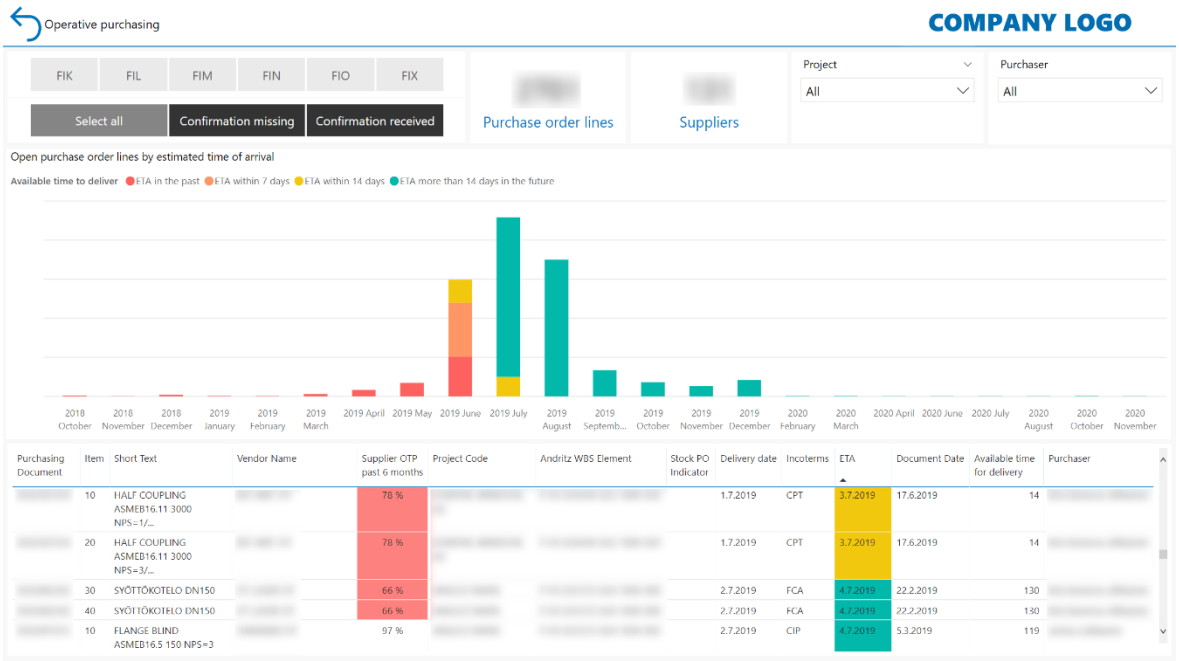


Figure 15. Open purchase orders analysis dashboard

The colors in the column chart denote how close the estimated delivery date of the purchase order is to the current day. Green means that the estimated delivery date is more than 2 weeks in the future, yellow that it is within 2 weeks, orange that it is within 1 week, and red that the estimated delivery date is already in the past. Clicking any of these sections of the column filters the data on the table to match the selection. For instance, clicking the yellow section on '2019 July' would only show purchase orders that are estimated to arrive in July, within two weeks from the current date.

The table in figure 15 employs some conditional formatting rules to assist in picking out useful information. Of particular interest are the ‘Supplier OTP past 6 months’ and ‘ETA’ fields in the table. Any supplier OTP below ANDRITZ Savonlinna Works Oy’s target of 95% is displayed in yellow, and OTP values below 85% are displayed in red. This immediately draws attention to purchase order rows where the supplier has been struggling to deliver on time in the past 6 months, which is a red flag for any purchase orders with critical timing requirements. The ‘ETA’ field in the table is color-coded to match the column diagram, so that green means the estimated delivery date is more than two weeks away, and yellow means that it is within two weeks of the current day, just like in the column diagram.

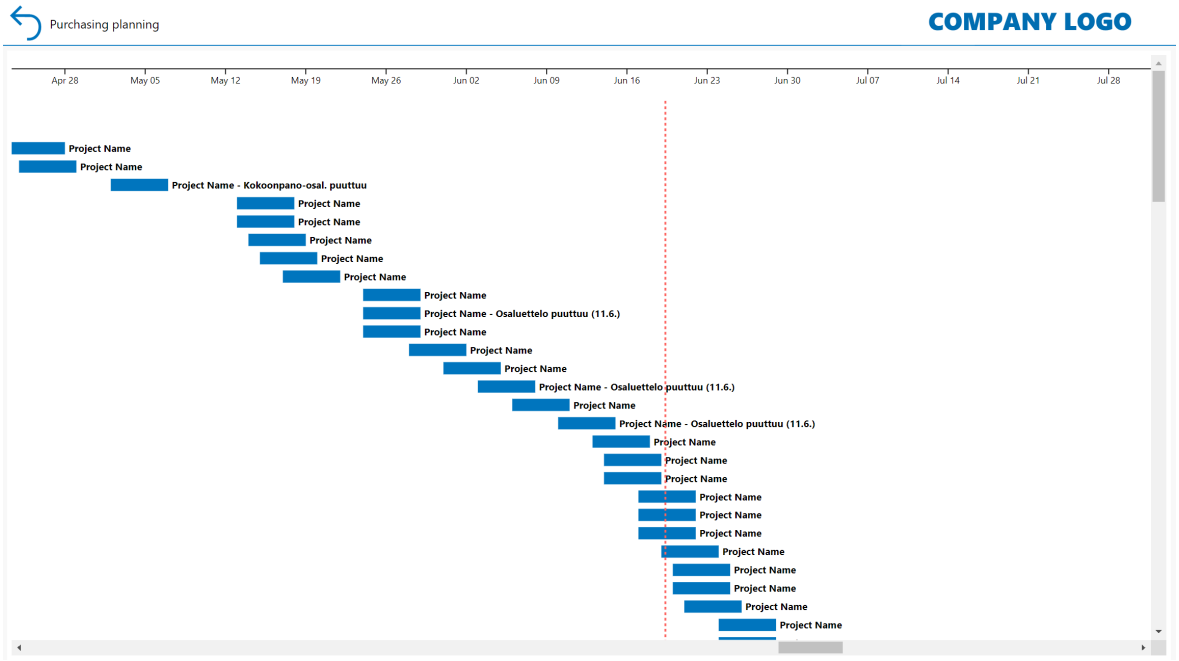


Figure 16. Purchasing planning schedule dashboard

Figure 16 depicts a Gantt chart of the scheduled workload of the purchasing planning process. The chart only displays tasks that aren’t finished, so any bars that fall behind the red ‘today’ line are past due. Users can leave comments on a SharePoint list that appear next to the project name on this dashboard, providing explanations for any tasks that are lagging behind or are at risk of doing so. Its purpose is to assist in prioritizing the tasks performed by the purchasing planning process, as well as provide early warning about projects that may experience delays in later processes of the procurement chain.

Table 3 introduces the full list of dashboards developed during the thesis, identifying the procurement process they are associated with, their main use case, the central KPIs featured in each dashboard, as well as any additional interactive features that assist in analyzing the data presented in the dashboard. It is intended to outline the full scope of the results of development beyond the dashboards included in the showcase. In addition to the 20 dashboards listed in this table, the full Power BI report included 6 navigation pages, 3 drill-through pages, and 1 diagnostics page for viewing when each table in the data model was last updated, bringing the total to 30 pages.

Table 3. Full list of dashboards developed during the thesis

Dashboard	Procurement overview
Process	Management
Use case	Evaluating the current status of the entire procurement process at a glance. Presented at weekly procurement meeting.
KPIs	Supplier on-time performance, purchase order volume, item arrival quantity, continuous improvement issues, open purchase requisitions queue, purchase order items created quantity, purchase order items by estimated time of arrival quantity, availability to production.
Interactivity	Drill-through to purchase order details table.
Dashboard	SQDCM (Safety, Quality, Delivery, Cost and Morale)
Process	Management
Use case	Presented as procurement's portion of the company's SQDCM board, a lean manufacturing tool, in weekly meetings.
KPIs	Supplier on-time performance, average time from goods arrival to receiving in days, item arrivals quantity, purchase order volume, purchase order items by estimated time of arrival quantity.
Interactivity	Drill-through to purchase order details table.

Table 3. Full list of dashboards developed during the thesis (continued)

Dashboard	Comprehensive management team meeting #1
Process	Management
Use case	Presented at monthly management team meetings. Page 1 of 5. Few visualizations per page to maximize readability during presentations.
KPIs	Supplier on-time performance, quantity of received purchase order items.
Interactivity	None.

Dashboard	Comprehensive management team meeting #2
Process	Management
Use case	Presented at monthly management team meetings. Page 2 of 5. Few visualizations per page to maximize readability during presentations.
KPIs	Top suppliers by purchase order volume and percentage share of total purchase order volume, quantity of purchase orders by millions of €.
Interactivity	None.

Dashboard	Comprehensive management team meeting #3
Process	Management
Use case	Presented at monthly management team meetings. Page 3 of 5. Few visualizations per page to maximize readability during presentations.
KPIs	Net order value by country.
Interactivity	None.

Table 3. Full list of dashboards developed during the thesis (continued)

Dashboard	Comprehensive management team meeting #4
Process	Management
Use case	Presented at monthly management team meetings. Page 4 of 5. Few visualizations per page to maximize readability during presentations.
KPIs	Quantity of reclamations by supplier, required compensation by supplier, most common defect problem codes, most common defect locations.
Interactivity	None.

Dashboard	Comprehensive management team meeting #5
Process	Management
Use case	Presented at monthly management team meetings. Page 5 of 5. Few visualizations per page to maximize readability during presentations.
KPIs	Engineering on-time performance, purchasing planning on-time performance, supplier on-time performance, average goods receiving response time, availability to production.
Interactivity	None.

Dashboard	Hot project schedule
Process	Strategic procurement
Use case	Analyzing prospective future projects
KPIs	Gantt chart displaying prospective future project schedules
Interactivity	Filtering by equipment group.

Table 3. Full list of dashboards developed during the thesis (continued)

Dashboard	Strategic KPIs
Process	Strategic procurement
Use case	Measuring the internal performance of the strategic procurement process.
KPIs	Supplier agreements, budget coverage, budget savings, budget overrun, quantity of purchase order items by millions of €.
Interactivity	Filtering by project, drill-through to supplier agreement details.
Dashboard	Supplier performance
Process	Strategic procurement
Use case	Supplier performance evaluation by KPIs.
KPIs	Supplier on-time performance, net order value, quantity of purchase orders and items, acknowledgement of orders, hitrate.
Interactivity	Filtering by supplier, supplier group, or date.
Dashboard	Supplier reclamations
Process	Strategic procurement
Use case	Supplier evaluation by reclamations and defects.
KPIs	Quantity of reclamations, distribution of defects by problem code, distribution of defects by location, quality costs, required compensation.
Interactivity	Filtering by supplier, supplier group, or date.

Table 3. Full list of dashboards developed during the thesis (continued)

Dashboard	Supplier reliability
Process	Strategic procurement
Use case	Determining if there is a connection between a supplier's on-time performance and their hitrate.
KPIs	Scatter plot displaying suppliers on a supplier on-time performance x-axis and a hitrate y-axis.
Interactivity	Drill-through to purchase order details table. Filtering by purchasing group, year, or month.
Dashboard	Purchasing planning schedule
Process	Purchasing planning
Use case	Viewing projects that the purchasing planning process is currently working on, upcoming projects, and work that is overdue.
KPIs	Gantt chart displaying purchasing planning scheduled workload
Interactivity	Link to SharePoint to add comments to items on the schedule
Dashboard	Purchasing planning KPIs
Process	Purchasing planning
Use case	Measuring the internal performance of the purchasing planning process.
KPIs	Quantity of released purchase requisitions, quantity of drawing revisions, purchasing planning on-time performance.
Interactivity	None.

Table 3. Full list of dashboards developed during the thesis (continued)

Dashboard	Open purchase requisitions
Process	Operative purchasing
Use case	Analyzing the open purchase requisition queue, identifying purchase requisitions that need to be prioritized.
KPIs	Scatter plot displaying open purchase requisitions on a time in queue x-axis and a remaining time to deliver y-axis.
Interactivity	Drill-through to purchase requisition details table. Filtering by purchasing group, time in queue, and remaining time to deliver.

Dashboard	Open purchase orders
Process	Operative purchasing
Use case	Analyzing open purchase orders, identifying potential risks in upcoming deliveries and analyzing open purchase orders that are past due.
KPIs	Quantity of purchase order items by estimated time of arrival, supplier on-time performance, available time for delivery.
Interactivity	Drill-through to purchase order details table, filtering by purchasing group, order acknowledgement received or missing, project, or purchaser.

Dashboard	Supplier on-time performance
Process	Operative purchasing
Use case	Determining whether a link exists between supplier on-time performance and available time for delivery.
KPIs	Scatter plot displaying suppliers on a supplier on-time performance x-axis and an average available time for delivery y-axis.
Interactivity	Drill-through to purchase order details table, filtering by purchasing group, year, or month.

Table 3. Full list of dashboards developed during the thesis (continued)

Dashboard	Delayed purchase orders
Process	Operative purchasing
Use case	Analyzing purchase orders that arrived late and whether the available time for delivery affected the length of delay.
KPIs	Scatter plot displaying suppliers on a length of delay x-axis with an average available time for delivery y-axis.
Interactivity	Drill-through to purchase order details table, filtering by purchasing group, year, or month.
Dashboard	Internal logistics – Internal
Process	Internal logistics
Use case	Measuring the internal performance of the internal logistics process and the upcoming workload.
KPIs	Purchase order items by estimated time of arrival, reservations, quantity of receiving inspection reports.
Interactivity	Drill-through to purchase order details table.
Dashboard	Internal logistics - Public
Process	Internal logistics
Use case	Displaying the upcoming workload of the internal logistics process on a public screen.
KPIs	Purchase order items by estimated time of arrival.
Interactivity	Drill-through to purchase order details table.

To summarize, the developed BI solution included 7 management dashboards, 5 strategic procurement dashboards, 2 purchasing planning dashboards, 4 operative purchasing dashboards, and 2 internal logistics dashboards. While the focus of the thesis was developing tools to assist in day-to-day operative tasks, some management dashboards that are mainly used in weekly or monthly meetings were also implemented, because the data required to build them was present in the data model, and the goal was to provide tools for every

procurement process in one place. Considering the number of dashboards by process, the importance of management seems overstated at 7 dashboards out of 20. However, this number is inflated by the presence of the comprehensive management team meeting dashboards, which were broken up to 5 pages to better accommodate being viewed from a distance during meetings by increasing the sizes of each individual visualization. The list of KPIs in each dashboard was included to provide additional context for the kind of information that can be gleaned from each dashboard besides the main use case description.

3.4 The data model in the finished BI solution

During development, the initial plan was to consolidate the data from all sources in an on-premises Microsoft SQL Server data warehouse by implementing an ETL system using SQL Server Integration Services (SSIS). By scheduling a script to run daily to execute a SAP GUI macro that exports the data from the ERP system into flat files on disk, the SQL Server Agent could be scheduled to execute an SSIS package that extracts the data from the flat files, performs the necessary transformations, and loads it into the data warehouse. The advantage of this approach is that it enables building a live connection to the data warehouse from Power BI and circumvents the need to import it into the data model or schedule data refreshes. Before the proof of concept status of the project was confirmed, it still looked likely that the Power BI cloud service could be used as the platform to deliver the BI solution to the end users. ANDRITZ company data wouldn't be stored in the Microsoft cloud platform if the data model is built on a live connection, which could have solved a potential issue when applying for permission to publish the BI solution to the Power BI cloud service.

However, the plan to build a data warehouse was eventually replaced with a self-service BI approach: connecting to the source data systems directly from Power BI Desktop. This decision was made to simplify the BI solution: since everything is in one place, it is easier to perform troubleshooting in the case of errors, and it also sped up the development process. The ETL process used in the final solution can be seen in figure 18. The report containing the dashboards and the data model is a Power BI file (*.pbix file format) that is stored on-premises and distributed to test users of the final proof of concept BI solution to be used with a local Power BI Desktop installation.

The source data for the completed BI solution is extracted from three different systems: SAP, SharePoint, and Turvallinenyritys.fi, and the data model it forms consists of 14 facts and 11 dimensions, as well as 6 other entities that contain no data but are used to organize the DAX measures containing the KPI calculations. The data model is designed according to a star schema, in that there are no relationships between dimensions. However, when viewing the data model diagram, it becomes apparent that many facts have no relationships at all: for all the entities in the data model, there are only 9 relationships between them prior to performance optimizations. There is also a case where a relationship is formed between two facts. This indicates that certain entities in the data model could benefit from a greater degree of normalization, while in the case of the related fact tables, denormalization or finding some way to combine the tables would improve the data model. The structure of the data model prior to optimization can be seen in figure 17. More detailed views of the data model where there is room for improvement can be found in the next chapter.

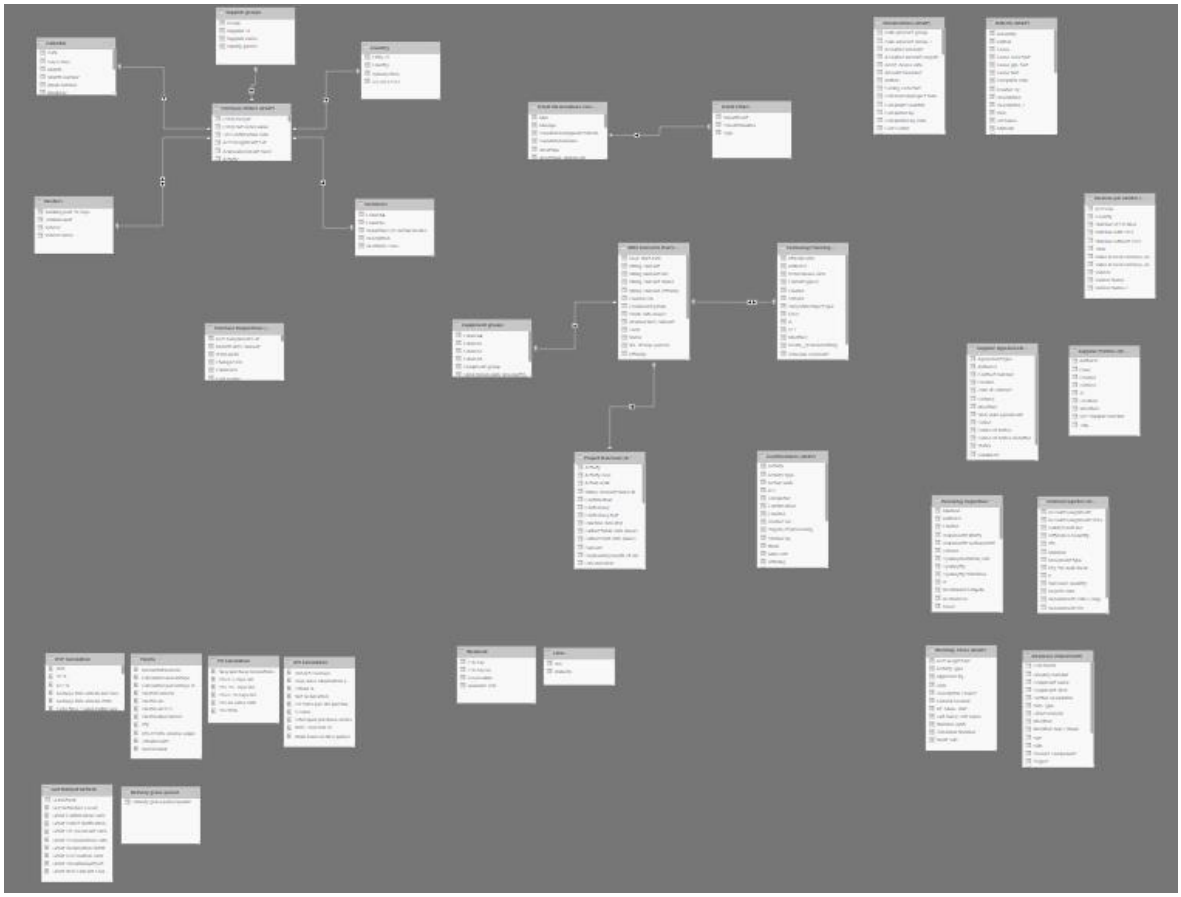


Figure 17. Overall structure of the data model in the finished BI solution

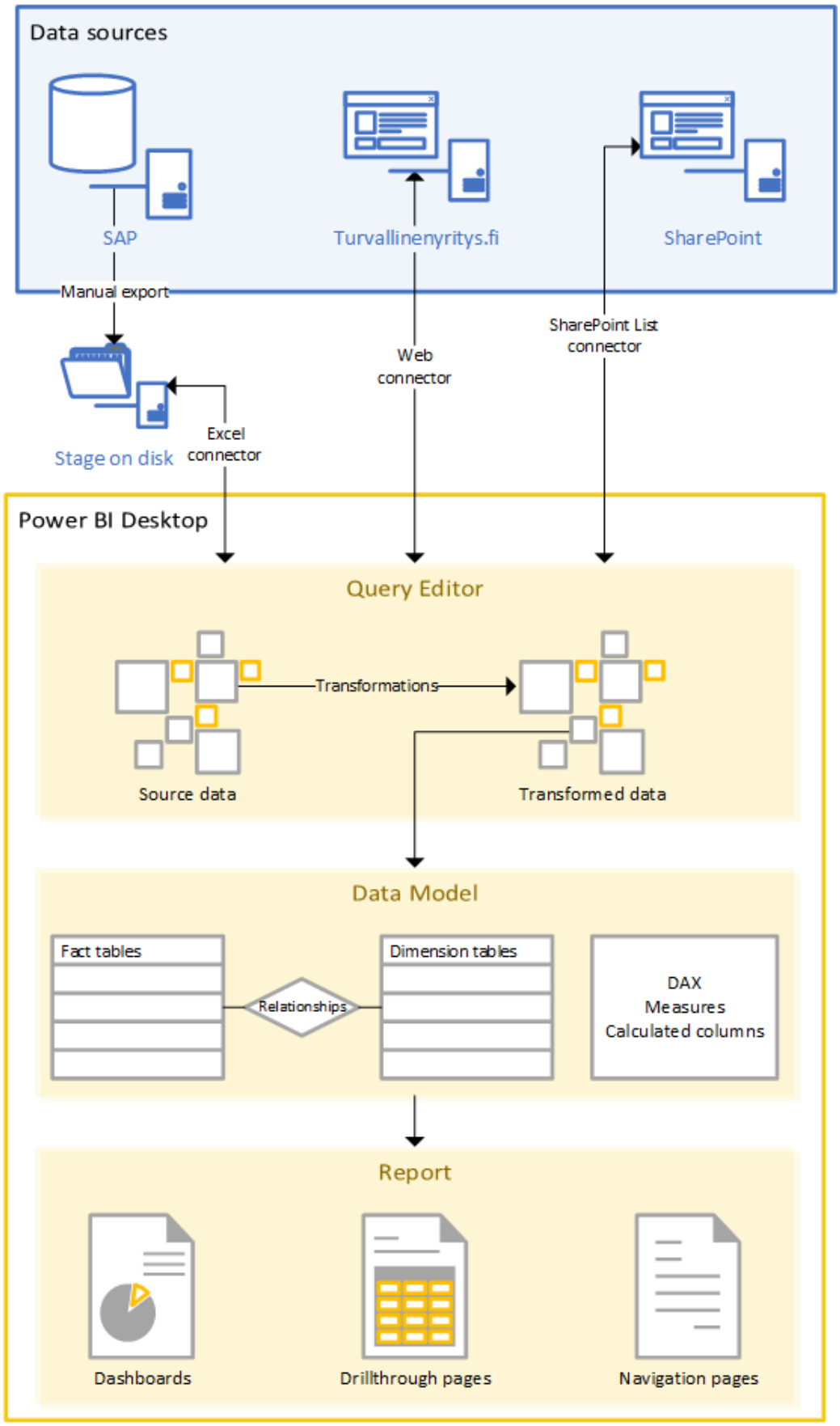


Figure 18. ETL process diagram of the developed BI solution

4 OPTIMIZATION AND ANALYSIS

In this chapter, the performance of the BI solution developed during this thesis will be analyzed using the performance analysis tool in Power BI Desktop, and the performance optimization methods introduced in the literature review will be tested and their impact on the performance measured.

4.1 Locating performance issues

In the literature review chapter of this thesis, the concept of system response time was introduced. In an SRT evaluation framework proposed by Doherty and Sorenson, the work flow of end users was determined not to be disrupted if SRT was kept below 10 seconds, if the system gives adequate feedback on its progress. [27] For the BI solution developed during this thesis, the goal is to keep SRT below 5 seconds for every task, because Power BI doesn't always offer adequate feedback to the user of the progression of its processing. To determine which dashboards are at risk of having performance issues and exceeding the 5 seconds SRT, the Power BI performance analyzer tool was used to measure the time it takes for all visualizations to load on each page of the report. The results can be seen in table 4.

Table 4. Preliminary performance analysis results

Dashboard	System response time
Procurement overview	1265 ms
SQDCM	792 ms
Management team meeting #1	215 ms
Management team meeting #2	229 ms
Management team meeting #3	132 ms
Management team meeting #4	235 ms
Management team meeting #5	179 ms
Hot project schedule	1076 ms
Strategic KPIs	264 ms
Supplier performance	423 ms
Supplier reclamations	431 ms
Supplier reliability	277 ms
Purchasing planning schedule	649 ms
Purchasing planning KPIs	225 ms
Open purchase requisitions	360 ms
Open purchase orders	337 ms
Supplier on-time performance	216 ms
Delayed purchase orders	246 ms
Internal logistics - Internal	228 ms
Internal logistics - Public	140 ms

The results of the preliminary analysis indicate that all dashboards are fully loaded well within the attentive SRT categories, most being fully loaded in less than a second. However, two dashboards stand out: Procurement overview and Hot project schedule. These are the only dashboards where the SRT exceeds one second. This was to be expected, because both dashboards display data from the largest fact tables in the data model: the purchase orders table in the case of the Procurement overview dashboard, which contains 31571 rows, and the projects table in the case of the Hot projects schedule dashboard, which contains 27963 rows. The Procurement overview dashboard also has the most visualizations out of all the dashboards in the report, totaling 13 different visualizations on the same page. While it may seem that any performance optimizations will result in little benefit given that the SRT values are already well within acceptable ranges, it should be noted that performance may be worse on end users' devices.

In light of the preliminary analysis, the optimization efforts will be focused on the purchase orders table and the visualizations that are slow to load in the Procurement overview dashboard to maximize the impact of the optimization efforts.

4.2 Analyzing the data model

During development, the logic used to build the data model was to include all fields that even had a remote chance of being useful. As a result, there were numerous unused columns in the purchase orders table. As stated during the literature review, the simplest method of optimizing a data model is removing unused fields, so that is a good place to start.

The purchase orders table had 85 columns prior to optimization, and after removing the columns that were unused, only 29 were left. Some of the removed columns had a very high cardinality, including one that had 29490 distinct values. Another change that was made was changing one string ID field to integer data format. This is considered a good practice, but can't be applied in every case, because sometimes leading zeroes are used, and those are lost when converting to integer, which can be a problem. Performing these changes reduced the size of the report file on disk from 26175 kilobytes to 25464 kilobytes, a modest decrease of 3 %.

The purchase orders table in the finished data model seen in figure 19 has a balance of normalization and denormalization. The table has relationships to five dimensions, but certain fields that are also found elsewhere in the data model are present in the purchase orders table, such as attributes related to the project, supplier, and delivery address that could have been normalized into dimension tables to create relationships between the purchase orders table and these other entities. However, because there was no need to create such relationships, it was decided to leave the fields in their denormalized state in the interests of performance and keeping the tables similar to the Excel files exported from the SAP GUI.

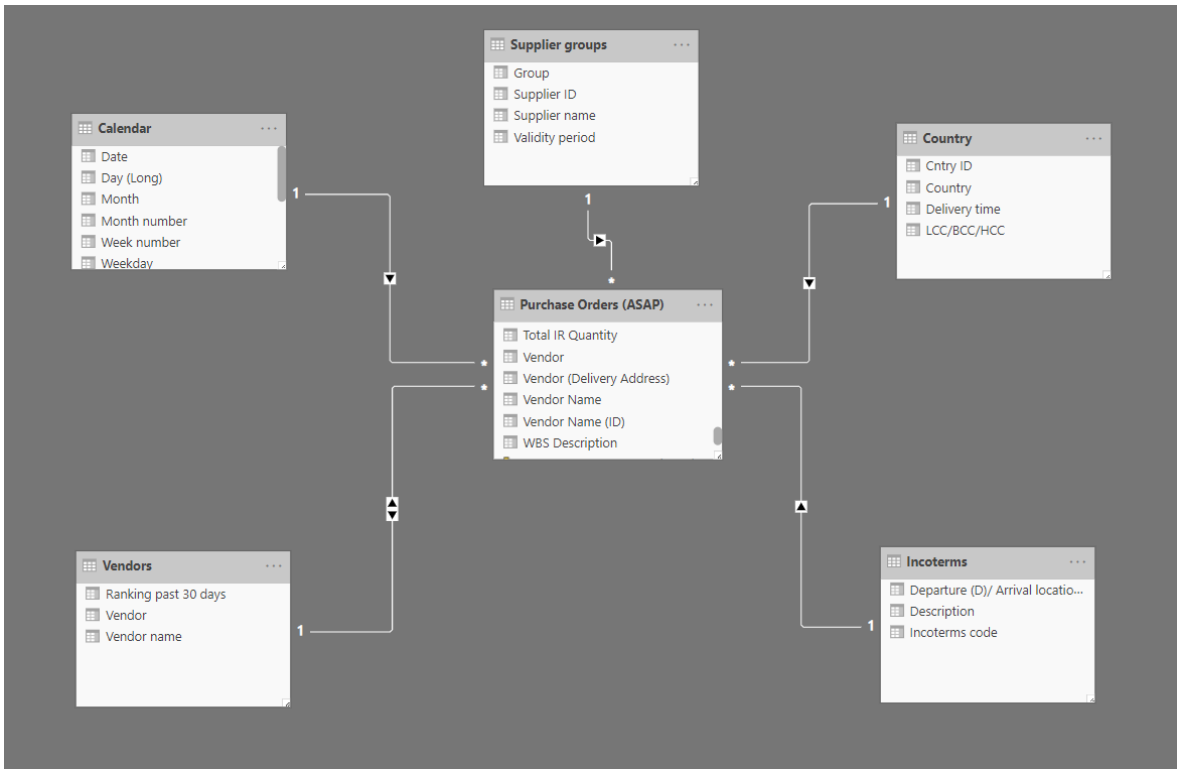


Figure 19. Purchase orders table in the data model of the finished BI solution

Upon closer inspection of the dimensions related to the purchase orders table, it was discovered that the calendar table contained 376564 rows. This was due to an oversight in how the calendar table was implemented: it was created using the DAX CALENDARAUTO function, which creates a date column automatically based on the earliest and latest date found in the data model. Due to an error, it created a date range from the year 1997 to 3027, with a very high cardinality where every value in the column was distinct. After adjusting the date range to match the purchase orders table's minimum and maximum dates, the

number of rows in the calendar table was reduced to 891, and the size of the report file on disk was reduced from 25464 kilobytes to 15223 kilobytes, a significant decrease of 40 %.

4.3 Analyzing the visualizations

Using the Power BI performance analyzer tool, it was determined that the two slowest loading visualizations on the Procurement overview dashboard were the Supplier on-time performance (last week) and Item arrivals (last week) visualizations, shown in figure 20, fully loading in roughly 1200 ms on average. This was unexpected, because the data displayed in these visualizations is filtered to just one week, and the 6 months rolling average supplier on-time performance visualization on the same page was loading considerably faster, in 500 ms on average, despite including data from the past 16 months as well as using an iterative DAX function to calculate its value.

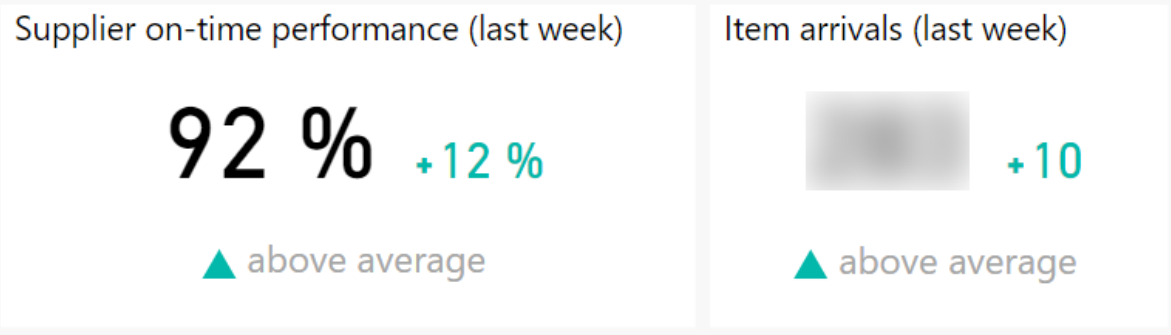


Figure 20. The slowest visualizations to load on the Procurement overview dashboard.

To find the cause of this slowdown, the visualizations in question should be broken down to the components that form them: the value being displayed, the value it is being compared to, and the visualization type. Considering that the item arrivals visualization loads just as slowly as the supplier on-time performance visualization, the value being displayed can be ruled out as the main cause of the performance dip. This is because in the supplier on-time performance visualization the value being displayed is a DAX measure, but in the item arrivals visualization it is a simple row count, which should be more efficient than a DAX calculation. This brings the comparison value into question: in both cases, it is a DAX measure calculating the year-to-date average for the displayed value. However, removing the comparison value from the visualization altogether doesn't affect its load time in any

significant way. This process of elimination leaves only one variable that can explain the slow loading time: the visualization type.

In this case, the visualization type is one that is not included in Power BI Desktop by default, but rather developed by a third party and imported to the report file from Microsoft AppSource [33]. The custom visualization type in question is called 'Card with States by OKViz' [34], and it provides some additional functionality over the default card visualization in Power BI in the form of displaying a secondary value next to the primary value being displayed on the card, as well as a customizable ticker. Replacing the visualization with Power BI's default card visualization results in the visualizations fully loading in roughly 500 ms, a considerable improvement over the 1200 ms load time with the custom card visualization. However, because the SRT is still well within non-disruptive ranges, the added features of the custom card visualization outweigh the performance benefits of the default version.

The other dashboards with longer than average SRTs, Hot project schedule and Purchasing planning schedule, also use custom visualizations. In their case, they use a custom Gantt chart visualization, because Power BI does not have a Gantt chart visualization by default. All the other dashboards are built exclusively with Power BI's default visualization types. If performance is a concern, custom visualizations should be avoided for best results.

4.4 Measuring the effectiveness of the performance optimizations

The performance optimization steps that were used to achieve these results are as follows: 1. 56 unused columns were removed from the purchase orders table, 2. a string ID column was converted to integer in the purchase orders table, 3. an error was removed from the definition of the calendar table, removing 375673 rows from the table, 4. the custom card visualization on the Procurement overview dashboard was replaced with Power BI's default card visualization. These steps reduced the file size of the Power BI report on disk from 26177 kilobytes to 15223 kilobytes, a decrease of 42 %. The memory usage of the Power BI report file as measured by the Windows Task Manager decreased from 2250 megabytes to 1950 megabytes, a decrease of 13 %.

The most substantial reduction of the Power BI report's file size was achieved by fixing the unfortunate error in the calendar table. The number of columns in the calendar table remained the same, but the number of rows was greatly reduced. In comparison, the purchase orders table kept the same number of rows, but its column count was considerably reduced. While the optimizations performed on the calendar table had a much greater effect on file size on disk, reducing the number of columns in the purchase order table did have a considerable effect on memory usage, going from 2250 megabytes to 2050 megabytes without the calendar table optimizations, a decrease of 9 %.

Performing SRT measurements with the Power BI performance analysis tool with all the optimizations in place resulted in slightly improved load times nearly across the board, as seen in table 5. However, the load times varied by approximately 5 % every time the visualizations were reloaded, so the only truly significant improvement in SRT was in the Procurement overview dashboard where the problematic custom card visualization was replaced with the default Power BI card visualization.

Table 5. Optimized performance analysis results

Dashboard	SRT – Preliminary	SRT - Optimized	Change%
Procurement overview	1265 ms	522 ms	- 59 %
SQDCM	792 ms	722 ms	- 9 %
Management team meeting #1	215 ms	208 ms	- 3 %
Management team meeting #2	229 ms	189 ms	- 17 %
Management team meeting #3	132 ms	104 ms	- 21 %
Management team meeting #4	235 ms	237 ms	1 %
Management team meeting #5	179 ms	156 ms	- 13 %
Hot project schedule	1076 ms	940 ms	- 13 %
Strategic KPIs	264 ms	234 ms	- 11 %
Supplier performance	423 ms	318 ms	- 25 %
Supplier reclamations	431 ms	363 ms	- 16 %
Supplier reliability	277 ms	252 ms	- 9 %
Purchasing planning schedule	649 ms	512 ms	- 21 %
Purchasing planning KPIs	225 ms	227 ms	1 %
Open purchase requisitions	360 ms	294 ms	- 18 %
Open purchase orders	337 ms	390 ms	15 %
Supplier on-time performance	216 ms	209 ms	- 3 %
Delayed purchase orders	246 ms	252 ms	2 %
Internal logistics - Internal	228 ms	170 ms	- 25 %
Internal logistics - Public	140 ms	105 ms	- 25 %

5 LESSONS LEARNED AND DISCUSSIONS

The goal of this thesis was to develop a business intelligence solution to assist in day-to-day supply chain management tasks for the procurement department at ANDRITZ Savonlinna Works Oy. To guide the empirical process, the theoretical section of the thesis was built on the following research questions: 1. “What are the technical approaches and design choices that could be applied to building an effective data model for a business intelligence solution from multiple data sources?” and 2. “What are the most significant factors that affect the efficiency of a business intelligence solution from a technical performance standpoint?”

The literature review served to establish a clear path from connecting to the source data to the specifics of building a data model with a plethora of examples. In this sense, the thesis is more successful at answering research question 1 than 2. The performance optimization steps introduced in the literature review are very surface level by comparison. In the end, however, the literature review supports the empirical section of the thesis, and all the elements introduced in it come to the forefront at some point during the later sections of the thesis.

The business intelligence solution developed as part of this thesis is a proof of concept. It is not an officially implemented and supported tool, and it is only used by a small group of test users, completely enclosed in the ANDRITZ ecosystem. Designed with the day to day operative tasks of supply chain management in mind, it offers a perspective of the business requirements of a single workshop, something which may be missing from business intelligence tools developed higher up in the hierarchy of a large multinational corporation. This perspective may prove valuable when developing Group procurement level business intelligence in the future to better serve end users. Going forward from finalizing the proof of concept stage of this project, the next step is to analyze what makes it different from the BI tools that are already available in the company. The number of different tools and technologies in use needs to be kept in check: there only needs to be one version of the truth, and several competing solutions within an organization will only serve to confuse and distract. Therefore, a likely future development path for the BI solution developed in this thesis is to have its features implemented in an official tool, eventually becoming obsolete. However, until then, this proof of concept will serve the needs of the end users. Since this

thesis only included the development of this BI solution, very little feedback from the end users has been collected. To better serve the needs of developing Group procurement's tools, the proof of concept needs some time in the user testing phase to collect valuable feedback now that development has reached the point of a full functionality.

The main goal of the thesis was met, and a fully functional business intelligence solution proof of concept was developed. However, as much of the development process preceded the theoretical work, some of the data modeling concepts introduced in the literature review were not fully observed during development. This served the purposes of the optimization section of the thesis, since there would potentially be greater performance gains to be had. The optimization steps were selected to provide different results: removing unused columns would affect granularity, changing data formats would affect the underlying way data is stored in the database engine, removing rows would make for a good comparison to removing columns, and optimizing the visualizations might show a change in a different metric than data model changes. Unfortunately, the varying scale of the changes makes the results difficult to compare to each other: the calendar table mishap ended up affecting too many rows, overshadowing more commonly useful optimizations. On the other hand, changing the data format from string to integer only affected a single column, so it ended up having a negligible effect. In addition to this, there were some challenges measuring the memory usage and SRT of the report, since both values fluctuated. To make the SRT measurements more comparable, Power BI Desktop was completely restarted before each measurement pass, so that no queries were cached in the database engine.

A valuable lesson learned during the development process was the effectiveness of face to face development meetings when working with a development tool that allows for rapid prototyping such as Power BI Desktop. Were it not for the development meetings in May and June, the end result would look very different, with much more focus on static KPI dashboards. Also, when working with an iterative development process in a project such as this, it is unwise to attempt to make a specification that is fully complete in the beginning, since new ideas will continuously come in and old ones will be put aside. The only dashboard that resembles its initial specification sheet is Procurement overview, and many others were completely redesigned as development progressed.

6 CONCLUSION

The business intelligence solution developed during this thesis consolidates data from multiple systems and provides improved tools for exploring and analyzing it to help procurement personnel at ANDRITZ Savonlinna Works Oy focus on the essential and work more efficiently. Having personal access to view this data gives direction to operative personnel: no longer having to wait for regularly scheduled KPI dashboards to be sent into their email or shown at meetings, instead being able to get rapid feedback on their performance whenever they try out something new or different. Given the scope of the project, the project organization that formed around it was vital to its success. The BI solution has provided more value to the company than expected in the beginning, benefiting from a shift in focus from KPI calculation to a more proactive set of analysis tools.

The next step for the result of this thesis is the validation testing phase and alignment with ANDRITZ Group procurement tools. There is still much potential for further developing business intelligence offerings at ANDRITZ Savonlinna Works Oy: other departments require dashboards just as much as procurement, and their use cases may be more suited for piloting a Power BI cloud service that is not yet fully supported in the organization due to the type of data to be visualized.

The performance optimization methods presented in this work were found to be somewhat effective, reducing the size of the report file on disk, its memory use, and system response times by non-negligible amounts. However, the theory surrounding data model optimization presented in this thesis was very surface-level, and the only method presented for troubleshooting performance issues was Power BI specific. Further research in this area would be required to gain a better understanding of general data model optimization methods.

In the future, business intelligence analysis will increasingly move from manual analysis to artificial intelligence and machine learning applications based on data mining where the risks and areas with room for improvement in the supply chain are automatically identified. There is no one-size-fits-all solution for supply chain collaboration: the demand characteristics of items vary but are often managed with the same model [35]. Business intelligence

applications based on automatic pattern analysis to identify demand characteristics could be used to determine the best management model to apply in each scenario. Automatically categorizing items by their demand characteristics also enables developing inventory management systems that accurately predict and react to changing demand [36]. However, this requires a very large data set for the results to be accurate. For the foreseeable future, interactive data visualizations and the options they provide for exploring data will remain useful. Today's business intelligence services enable performing rapid analysis on data and sharing the findings within different levels of an organization in a faster and more convenient manner than ever before, making it possible to increase the level of formalization in supply chain innovation and unlock new potential for decreasing costs [37].

REFERENCES

[1] T. Van-Hau, "Getting value from Business Intelligence systems: A review and research agenda," *Decision Support Systems,* vol. 93, pp. 111-124, 2017.

[2] T.-P. Liang and Y.-H. Liu, "Research Landscape of Business Intelligence and Big Data analytics: A bibliometrics study," *Expert Systems with Applications,* vol. 111, pp. 2-10, 2018.

[3] R. Van der Meulen and C. Pettey, "Gartner Says Self-Service Analytics and BI Users Will Produce More Analysis Than Data Scientists Will by 2019," Gartner, Inc, 25 January 2018. [Online]. Available: https://www.gartner.com/en/newsroom/press-releases/2018-01-25-gartner-says-self-service-analytics-and-bi-users-will-produce-more-analysis-than-data-scientists-will-by-2019. [Accessed 15 June 2019].

[4] K. Božič and V. Dimovski, "Business intelligence and analytics for value creation: The role of absorptive capacity," *International Journal of Information Management,* vol. 46, pp. 93-103, 2019.

[5] T. Partala and T. Saari, "Understanding the most influential user experiences in successful and unsuccessful technology adoptions," *Computers in Human Behavior,* vol. 53, pp. 381-395, 2015.

[6] Microsoft, "Power BI | Interactive Data Visualization BI Tools," [Online]. Available: https://powerbi.microsoft.com/en-us/. [Accessed 12 6 2019].

[7] T. Lachev, Applied Microsoft Power BI: Bring your data to life, Prologika Press, 2017.

[8] Tableau Software, "Business Intelligence and Analytics Software," [Online]. Available: https://www.tableau.com/. [Accessed 15 June 2019].

[9] SAP SE, "SAP Lumira Discovery," [Online]. Available: https://saplumira.com/. [Accessed 15 June 2019].

[10] SAP SE, "SAP Analytics Cloud | End-to-end Analytics for the Intelligent Enterprise | SAP," [Online]. Available: https://www.sapanalytics.cloud/. [Accessed 15 June 2019].

[11] Qlik, "A Complete Business Intelligence (BI) Platform | Qlik," [Online]. Available: https://www.qlik.com/us/products. [Accessed 15 June 2019].

[12] Gartner, Inc, "Best Business Intelligence and Analytics Software of 2018 as Reviewed by Customers," November 2018. [Online]. Available: https://www.gartner.com/reviews/customers-choice/analytics-business-intelligence-platforms. [Accessed 15 June 2019].

[13] G2, Inc, "Best Business Intelligence Software," [Online]. Available: https://www.g2.com/categories/business-intelligence. [Accessed 15 June 2019].

[14] Microsoft, "Power BI Desktop--Interactive Reports | Microsoft Power BI," [Online]. Available: https://powerbi.microsoft.com/en-us/desktop/. [Accessed 13 6 2019].

[15] J. Sundström, "Microsoft Power BI Dataflows – Data warehousing made simple?," Capacent AB, 27 March 2019. [Online]. Available: https://capacent.com/sv/about/news/2019/microsoft-power-bi-dataflows-data-warehousing-made-simple/. [Accessed 14 June 2019].

[16] V. Rainardi, Building a Data Warehouse: With Examples in SQL Server, Apress, Inc, 2008.

[17] F. Riggins and B. Klamm, "Data governance case at KrauseMcMahon LLP in an era of self-service BI and Big Data," *Journal of Accounting Education,* vol. 38, pp. 23-36, 2017.

[18] M. Llave, "Data lakes in business intelligence: reporting from the trenches," *Procedia Computer Science,* vol. 138, pp. 516-524, 2018.

[19] R. Salem and A. Abdo, "Fixing rules for data cleaning based on conditional functional dependency," *Future Computing and Informatics Journal,* vol. 1, no. 1-2, pp. 10-26, 2016.

[20] Microsoft, "Data sources in Power BI Desktop," 10 June 2019. [Online]. Available: https://docs.microsoft.com/en-us/power-bi/desktop-data-sources. [Accessed 16 June 2019].

[21] Microsoft, "Quick tour of the Power Query M formula language," 12 December 2018. [Online]. Available: https://docs.microsoft.com/en-us/powerquery-m/quick-tour-of-the-power-query-m-formula-language. [Accessed 16 June 2019].

[22] A. Ferrari and M. Russo, Analyzing Data with Microsoft Power BI and Power Pivot for Excel, Redmond: Microsoft Press, 2017.

[23] S. Allen and E. Terry, Beginning Relational Data Modeling, Apress, Inc, 2005.

[24] Mockaroo, LLC, "Mockaroo - Random Data Generator and API Mocking Tool | JSON / CSV / SQL / Excel," 2019. [Online]. Available: https://mockaroo.com/. [Accessed 22 June 2019].

[25] A. Ferrari and M. Russo, The Definitive Guide to DAX: Business intelligence with Microsoft Excel, SQL Server Analysis Services, and Power BI, Redmond: Microsoft Press, 2015.

[26] Microsoft, "DAX function reference," 19 April 2019. [Online]. Available: https://docs.microsoft.com/en-us/dax/dax-function-reference. [Accessed 23 June 2019].

[27] R. Doherty and P. Sorenson, "Keeping Users in the Flow: Mapping System Responsiveness with User Experience," *Procedia Manufacturing,* vol. 3, pp. 4384-4391, 2015.

[28] A. Cofsky, "Power BI Desktop May 2019 Feature Summary," Microsoft, 17 May 2019. [Online]. Available: https://powerbi.microsoft.com/en-us/blog/power-bi-desktop-may-2019-feature-summary/. [Accessed 17 June 2019].

[29] Microsoft, "Power BI performance best practices," 18 May 2018. [Online]. Available: https://docs.microsoft.com/en-us/power-bi/power-bi-reports-performance. [Accessed 17 June 2019].

[30] ANDRITZ Savonlinna Works Oy, "ANDRITZ Savonlinna Works Oy, Savonlinna, Finland," [Online]. Available: https://www.andritz.com/pulp-and-paper-en/locations/savonlinna-works-oy. [Accessed 19 June 2019].

[31] ANDRITZ Savonlinna Works Oy, "100 Years History of ANDRITZ Savonlinna Works," [Online]. Available: https://www.andritz.com/pulp-and-paper-en/locations/savonlinna-works-oy/history. [Accessed 19 June 2019].

[32] F. Cervone, "Understanding agile project management methods using Scrum," *OCLC Systems & Services: International digital library perspectives,* vol. 27, no. 1, pp. 18-22, 2011.

[33] Microsoft, "Business Apps - Microsoft AppSource," [Online]. Available: https://appsource.microsoft.com/en-us/marketplace/apps?product=power-bi-visuals. [Accessed 23 June 2019].

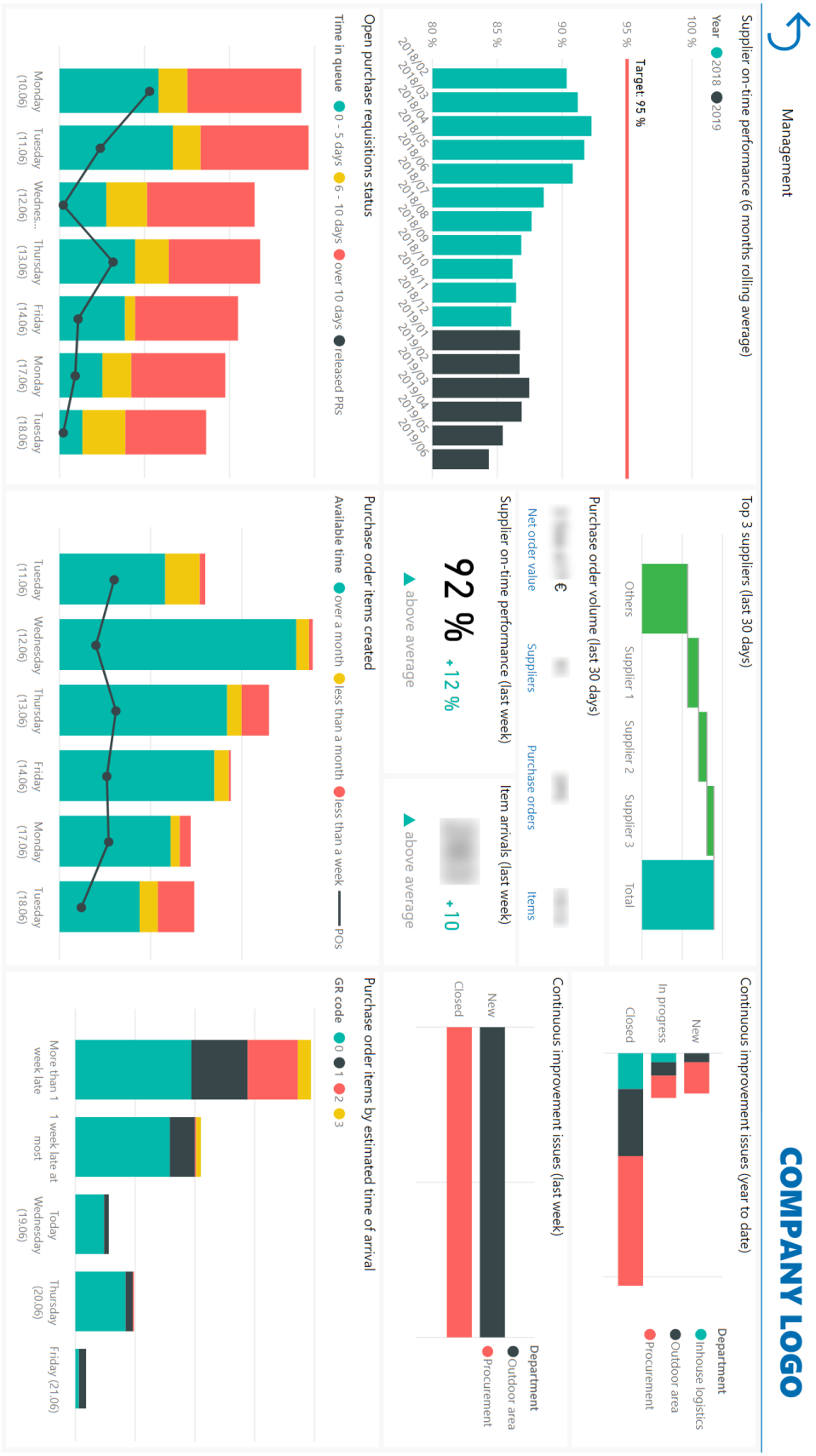
[34] OKViz, "Card with States by OKViz - Microsoft AppSource," 9 March 2017. [Online]. Available: https://appsource.microsoft.com/en-us/product/power-bi-visuals/WA104380967?tab=Overview. [Accessed 23 June 2019].

[35] E. Salmela, A. Happonen and J. Huiskonen, "Best collaboration practices in supply chain of technical wholesale items," *International Journal of Collaborative Enterprise,* vol. 2, no. 1, pp. 16-38, 2011.

[36] A. Happonen, "Adjusting inventories based on demand prediction using dynamic inventory balancing model," in *Technology Management for Emerging Technologies (PICMET)*, 2012.

[37] E. Salmela, C. Santos and A. Happonen, "Formalisation of front end innovation in supply network collaboration," *International Journal of Innovation and Regional Development,* vol. 5, no. 1, pp. 91-111, 2013.

APPENDIX 1. Dashboard examples



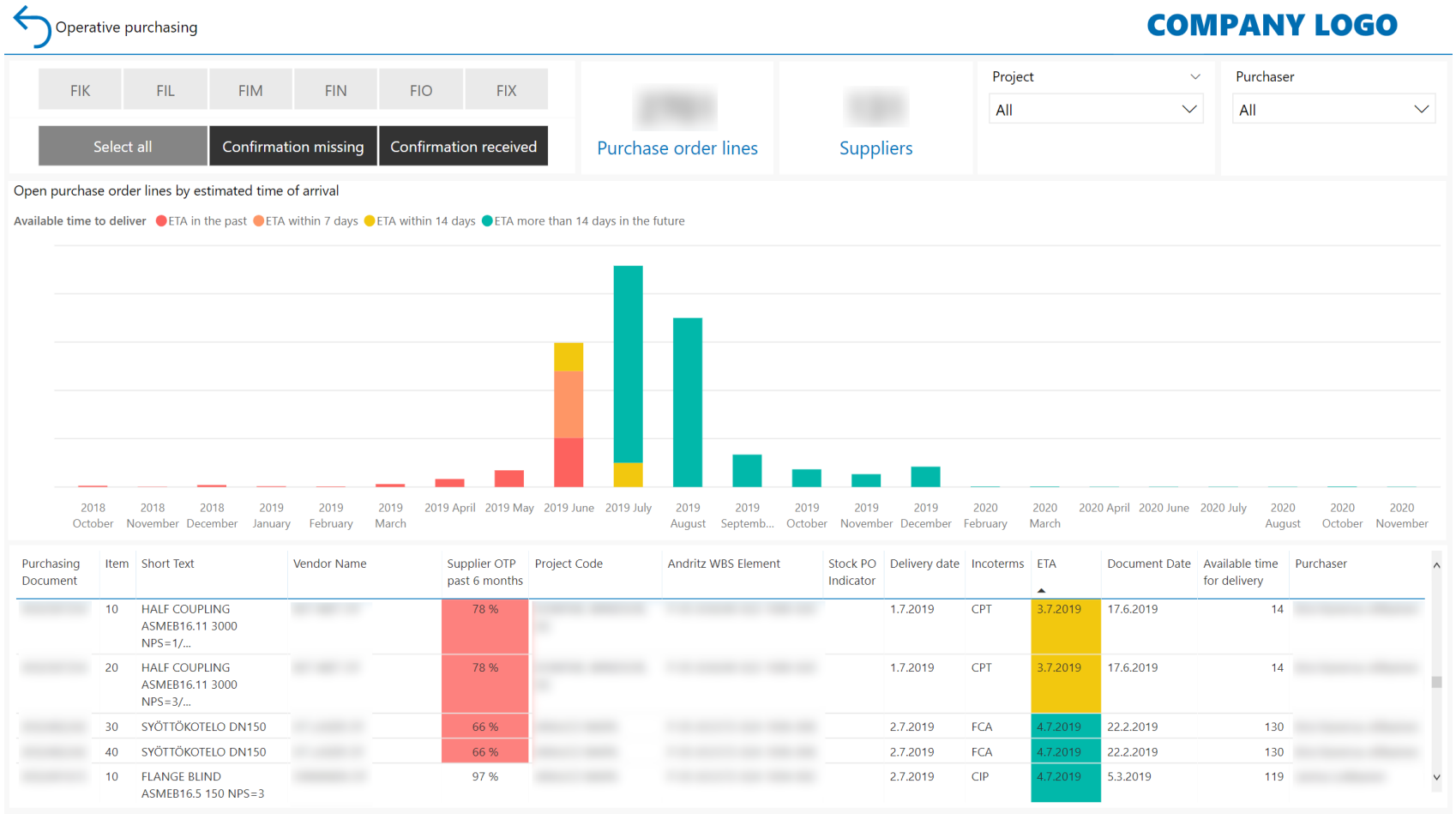
APPENDIX 1. (continues)

APPENDIX 1. (continues)

COMPANY LOGO

Purchase Requisition	Purchasing Group	Andritz WBS Element	Project Code	Remaining time to deliver	Time in queue	Delivery date	Material Description	Short Text	Release date
	FIL			-34	55	16.5.2019	PLATE L 888 W 942 MM		25.4.2019
	FIL			-34	55	16.5.2019	PLATE L 2016 W 2016 H 40 MM		25.4.2019
	FIL			-34	55	16.5.2019	PLATE L 1187.5 W 533 MM #2#		25.4.2019
	FIL			-34	55	16.5.2019	HALF COUPLING ASMEB16.11 3000 NPS=3/...		25.4.2019
	FIL			-34	55	16.5.2019	PLUG SQUARE ASMEB16.11 3000 NPS=1/2"...		25.4.2019
	FIL			-34	55	16.5.2019	TUBE_PIPE L 140 MM #2#		25.4.2019
	FIL			-34	55	16.5.2019	HALF COUPLING ASMEB16.11 3000 NPS=1/...		25.4.2019
	FIL			-34	55	16.5.2019	FLAT BAR L 248 W 50 H 6 T 6 MM		25.4.2019
	FIL			-34	55	16.5.2019	ROUND BAR L 766 D 16 MM #2#		25.4.2019
	FIL			-34	55	16.5.2019	SHEET L 1119 W 157 MM #2#		25.4.2019
	FIL			-34	55	16.5.2019	PLATE L 150 W 100 H 95 MM		25.4.2019
	FIL			-34	55	16.5.2019	PLATE L 1524 W 914 MM #2#		25.4.2019
	FIL			-34	55	16.5.2019	PLATE L 525 W 559 MM #2#		25.4.2019
	FIL			-34	55	16.5.2019	PLATE L 2142 W 1128.5 MM #2#		25.4.2019
	FIL			-34	55	16.5.2019	REINFORCEMENT PLATE L 944 W 944 H 35...		25.4.2019
	FIL			-34	55	16.5.2019	SHEET L 546 W 95 T 3 MM #2#		25.4.2019
	FIL			-34	55	16.5.2019	TUBE_PIPE L 685 D 21.3 MM #2#		25.4.2019
	FIL			-34	55	16.5.2019	PLUG SQUARE ASMEB16.11 3000 NPS=1/2"...		25.4.2019
	FIL			-34	55	16.5.2019	THREADED COUPLING L 48 W 40 H 40 MM #2#		25.4.2019
	FIL			-34	55	16.5.2019	PLATE L 203 W 154 MM #2#		25.4.2019
	FIL			-34	55	16.5.2019	PLATE L 1025 W 947 MM		25.4.2019
	FIL			-34	55	16.5.2019	SHEET L 1522 W 147 T 3 MM #2#		25.4.2019
	FIL			-34	55	16.5.2019	PLATE L 219 W 25 T 25 MM #2#		25.4.2019
Total									

APPENDIX 1. (continues)



Purchasing Document	Item	Short Text	Vendor Name	Supplier OTP past 6 months	Project Code	Andritz WBS Element	Stock PO Indicator	Delivery date	Incoterms	ETA	Document Date	Available time for delivery	Purchaser
[illegible]	10	HALF COUPLING ASMEB16.11 3000 NPS=1/...	[illegible]	78 %	[illegible]	[illegible]		1.7.2019	CPT	3.7.2019	17.6.2019	14	[illegible]
[illegible]	20	HALF COUPLING ASMEB16.11 3000 NPS=3/...	[illegible]	78 %	[illegible]	[illegible]		1.7.2019	CPT	3.7.2019	17.6.2019	14	[illegible]
[illegible]	30	SYÖTTÖKOTELO DN150	[illegible]	66 %	[illegible]	[illegible]		2.7.2019	FCA	4.7.2019	22.2.2019	130	[illegible]
[illegible]	40	SYÖTTÖKOTELO DN150	[illegible]	66 %	[illegible]	[illegible]		2.7.2019	FCA	4.7.2019	22.2.2019	130	[illegible]
[illegible]	10	FLANGE BLIND ASMEB16.5 150 NPS=3	[illegible]	97 %	[illegible]	[illegible]		2.7.2019	CIP	4.7.2019	5.3.2019	119	[illegible]

APPENDIX 1. (continues)