|  |
| --- |
| [Term Paper Title] |
| Assignment ReportB8IT104 - Data Warehousing and Business IntelligenceDecember 16, 2017 |

Contents

[Part 1: Business Drivers 2](#_Toc25684327)

[1.1 Background 2](#_Toc25684328)

[1.2 Subject Area for Analysis 2](#_Toc25684329)

[1.3 Strategy Map 3](#_Toc25684330)

[1.4 Balanced Scorecard 4](#_Toc25684331)

[1.5 Key Performance Indicators 4](#_Toc25684332)

[1.6 Vision & Goals for the Data Warehouse 6](#_Toc25684333)

[1.7 Key Stakeholders 6](#_Toc25684334)

[Part 2: Data Modelling 8](#_Toc25684335)

[2.1 Data Warehouse Schema 8](#_Toc25684336)

[2.2 Reasons for Design 9](#_Toc25684337)

[Part 3: Implement Tables and ETL Procedure 13](#_Toc25684338)

[3.1 Implementation using Microsoft SQL Server 13](#_Toc25684339)

[3.2 Motivation for using Microsoft SQL Server 18](#_Toc25684340)

[3.3 Comparison of Microsoft SQL Server with Apache Spark 18](#_Toc25684341)

[Part 4: Reporting and Analysis 20](#_Toc25684342)

[4.1 Dashboard Design 20](#_Toc25684343)

[4.2 Dashboard Analysis 28](#_Toc25684344)

[Appendix 1 – SQL Script to Build the Data Warehouse 33](#_Toc25684345)

[Appendix 2 - ETL Script to Populate the Data Warehouse 35](#_Toc25684346)

[Appendix 3 - ETL Data Validation Scripts 39](#_Toc25684347)

[References 40](#_Toc25684348)

# Part 1: Business Drivers

## 1.1 Background

The Northwind database contains sales data for a company named "Northwind Traders", a fictitious food trading company. The database captures sales transactions that occur between the Company and its customers as well as information about the products the Company trades in and the inventory it holds. The database also contains details relating to the Company's customers, suppliers and employees.

## 1.2 Subject Area for Analysis

For the purposes of this assignment, I will focus my analysis on **product assortment**.

Product assortment is critical to retail success. The right combination of products will maximize sales. Northwind Traders therefore needs to know how its current product lines are performing and which products, if any, are underperforming. This will allow the Company to make informed decisions as to whether certain products should be discontinued.

To stay competitive in the marketplace, Northwind Traders must also stay on top of the latest food trends and continually add new products. This will enable the Company to differentiate itself from its competitors, offer the most in-demand products and grow its customer base, thereby increasing sales revenue.

## 1.3 Strategy Map

The Strategy Map I have developed for Northwind Traders illustrates the main strategic goal for the Company and the objectives it has set itself in order to meet this goal;

**Fig 1**: Strategy Map for Northwind Traders

## 1.4 Balanced Scorecard

A Balanced Scorecard will help the Company to monitor progress against its strategic objectives and will also help Senior Management to define and manage action plans in support of these objectives.

The Balanced Scorecard I have developed for Northwind Traders based on the Strategy Map outlined in [Section 1.3](#StrategyMap) is detailed below;

|  |  |  |
| --- | --- | --- |
| Strategic Perspective | Business Objective | Key Performance Indicator |
| Financial | * Increase Sales Revenue * Maximise Product Sales | * Sales Growth Rate * Product Performance |
| Customer | * Increase Product Appeal | * Purchase Frequency |
| Internal Business Processes | * Fill Sales Orders on Time * Move Products Quickly | * Sales Orders Filled per Unit Time * Order Shipment Cycle Time |
| Learning and Growth | * Improve Focus on Sales | * Sales per Rep |

**Table 1**: Balanced Scorecard for Northwind Traders

## 1.5 Key Performance Indicators

The Key Performance Indicators that will be used to measure the Company’s performance against its strategic objectives are described in more detail below;

### Financial KPIs:

* **Sales Growth Rate**: Measures the pace at which the organization's sales revenue is increasing or decreasing.

|  |
| --- |
| Formula:  [(Sales for the current period - Sales for the previous period)/Sales for the previous period]x100 |

* **Product Performance**: Ranks products based on revenue performance to inform the sales team which products are selling well and which products need special attention.

|  |
| --- |
| Formula:  Products ranked in order of sales revenue for the current period |

### Customer KPIs:

* **Purchase Frequency**: Measures the number of times a customer makes a purchase within a given time period. This KPI helps to gain insight into customer behaviour and the appeal of products.

|  |
| --- |
| Formula:  (Number of orders placed in the specified time period)/Number of unique customers within the time period |

### Internal Business Processes KPIs:

* **Sales Orders Filled per unit Time**: Measures the average amount of sales orders filled within a specific time period. This metric provides insight into how quickly sales orders are being filled and if they are being done in a time-efficient manner.

|  |
| --- |
| Formula:  (Number of sales orders filled in time period A + #Orders in time B + #Orders in time C + #Orders in time N) / Total number of time units in the specific reporting period |

* **Order Shipment Cycle Time**: Measures the length of time between the order being placed and the shipment of the order. This KPI helps to gain insight into the efficiency of the supply chain process.

|  |
| --- |
| Formula:  (Number of days to ship order A + #Days to ship order B + #Days to ship order C + #Days to ship order N) / Total number of sales orders placed in the specific reporting period |

### Learning & Growth KPIs:

* **Sales per Rep**: Measures the performance of each of the sales reps. This KPI measures the ability of each of the sales reps to generate revenue for the organization.

|  |
| --- |
| Formula:  Total number of sales orders processed by each employee in the specified time period |

## 1.6 Vision & Goals for the Data Warehouse

The data warehouse will be used to produce reports and dashboards based on the KPIs identified in the Balanced Scorecard. This analysis will inform strategic decision making by the Company in relation to product assortment.

The goals of the data warehouse are as follows;

* It will present data that’s understandable to the Business users and serve as the basis for data-driven decision making
* It will present information consistently via the creation of a dashboard that will be used to monitor performance from one quarter to the next
* It will store data in a format that is easily adaptable to change
* It will deliver fast query performance and present information in a timely manner

## 1.7 Key Stakeholders

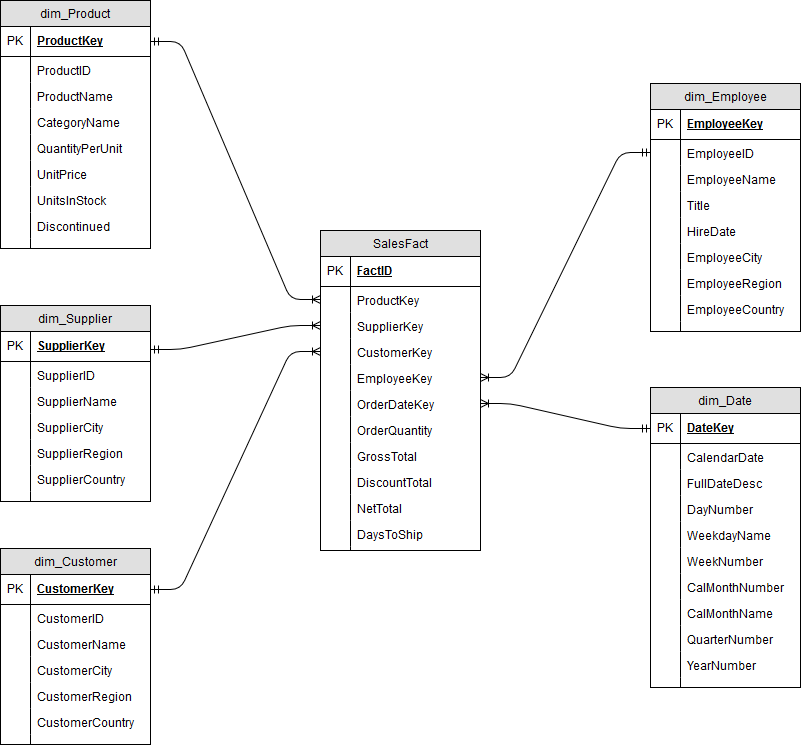
Key stakeholders for the project are as follows;

* **Senior Management**: The Senior Management of Northwind Traders regularly re-evaluates the Company’s product lines to see what's selling and what’s not. Reporting and analysis from the data warehouse will help guide their decisions in relation to product assortment.
* **Shareholders**: The right combination of products will maximize sales for the Company and thereby benefit shareholders. Whereas, the wrong decision can result in falling sales revenues for the Company and potentially even bankruptcy.
* **Customers**: Existing customers will be impacted by any decisions to drop a product. If a similar replacement product is not offered by the Company, then customers may take their business elsewhere. Conversely, if the Company adds new products that are appealing, this may help broaden their customer base.
* **Suppliers**: The Company’s suppliers will be impacted by any decisions made to drop a product, as they will lose the business they previously received from the Company.
* **Employees**: To help meet the strategic objectives of the Company, employees will be set clear performance objectives. Reporting from the data warehouse will allow Senior Management to measure employees’ performance against expected sales targets.

# Part 2: Data Modelling

## 2.1 Data Warehouse Schema

The data warehouse will be implemented using a star schema design which consists of one fact table surrounded by several dimension tables.

 **Fig 2**: Star Schema Design for Northwind Data Warehouse

## 2.2 Reasons for Design

The data warehouse schema was designed following Kimball’s four-step design process below;

* Select the Business Process
* Declare the Grain
* Identify the Dimensions
* Identify the Facts

### Select the Business Process

The dimensional model is required to measure business process events. In this case, there is just one event being measured; the placement of orders.

### Declare the Grain

The fact table will focus on the results of the business process. Since the dimensional model represents the most detailed, atomic information captured by the business process, the grain of the fact table will be one row per product ordered. This is a lower grain than a customer order, since a customer may order several products within a single order. An individual row in the fact table will therefore represent a *line item* of a customer order – ie. the fact table will have the same grain as the “Order Details” table in the Northwind operational database.

Seeing as each row in the fact table has a one-to-one relationship to the measurement event, and in this case there is only one event being measured, only one fact table will be required. For this reason, the data warehouse schema will be implemented using a star schema design.

### Identify the Dimensions

Dimensions were identified by asking the question, “How should data relating to the ordering of products be organized and presented?” Business users will want to know the “who, what, where, when, why and how” context surrounding the business process event. They will therefore have questions such as the following;

* Who bought the products?
* Who sold the products?
* What was sold?
* When was it sold?

The descriptive attributes needed to answer these questions are Customer, Employee, Product and Order Date. Having these dimensions in the schema will allow Business users to filter and group the facts by each of these attributes and therefore they will be able to obtain answers to the questions they have about the sales data.

Are there any other attributes available in the operational database that Business users may want to segment the sales data by? They may potentially want to know who supplied each of the products. This would allow them to then slice-and-dice the sales data by supplier. Since each product is tied to one and only one supplier, it will not generate any extra rows in the fact table to include the supplier. So, an additional dimension table has been added to the schema for Supplier and the SupplierKey has been added to the fact table. This will allow the sales data to be aggregated by an additional attribute.

### Identify the Facts

Lastly, the facts in the dimensional model were determined by answering the question “What is the business process measuring?” For each product ordered, several key pieces of information will be required – for example, the quantity of the product ordered, the gross total for the order, the discount total and the net total. These are all derived facts that will be calculated by the ETL process when the data is loaded from the operational database into the data warehouse. Although they are calculated facts, they will be physically stored in the data warehouse in order to ensure that they are computed consistently and mitigate the risk of user calculation errors in the BI application.

The number of days it took for the order to be shipped will also be calculated by the ETL process and stored in the data warehouse, since this metric is required to measure the KPI for Order Shipment Cycle Time. So, rather than storing the Shipped Date in the fact table and performing this calculation in the BI tool, the calculation will instead be performed by the ETL process and the result stored in the data warehouse. In this way, the data warehouse schema design simplifies the reporting logic, since these derived facts are all measures that will be aggregated in the reporting that is produced.

Some of the metrics that are required to measure the defined KPIs are percentages or ratios that will need to be computed using the BI tool because the calculations cannot be stored in the fact table (as they would violate the grain of the fact table). So, for example, Sales Growth Rate will need to be calculated using the BI tool. However, having the Net Total pre-calculated in the data warehouse will speed up query performance in the BI tool when calculating this metric.

Since it doesn’t make any sense to sum the unit price of a product across any of the dimensions (unit price is non-additive), the Unit Price has been omitted from the fact table. Freight has also been omitted from the fact table since none of the KPIs defined in [Section 1.5](#KPIs) involve the use of this field.

### Dimension Table Details

Fields that are not required for KPI reporting (such as phone, fax, address, photo, notes etc.) have been omitted from the dimension tables.

In addition, one-to-many hierarchies in the operational database have been flattened in the data warehouse schema design. So, for example, Category Name has been added to the Product dimension which will allow sales metrics to be rolled up at category level. This means that there will be duplicated data values in the Product dimension table (in this way, the star schema design is de-normalized). However, a key benefit of this design is that it is optimized for query performance and fast aggregation - since the data is flat, summary statistics can be calculated without having to perform a complex join.

The Date dimension table in the data warehouse allows Business users to summarize and analyze the data by various calendar grouping such as months, quarters or years. Having these attributes available in the Date dimension table means that Business users do not need to write SQL date functions in the BI tool in order to aggregate the data by these time periods. It also means that non-standard date attributes such as fiscal periods or seasons can easily be added to the Date dimension if required. This would provide the capability for users to slice and dice by these additional date groupings if required.

The dimension tables contain the ID fields from the Northwind operational database so that the ETL process can extract the data, transform it and load it into the fact table (the operational keys are needed to match the data from the Northwind operational database with the dimension tables in the data warehouse).

However, the data warehouse schema uses surrogate keys as unique identifiers for the dimension tables. This ensures that the data warehouse is insulated from any unexpected changes that might occur in the operational database, such as the deletion of a Product ID for a discontinued product. It also improves performance in the data warehouse, since the joins between the dimensions and the fact table are more efficient (surrogate keys take up less storage space and hence, the indexes are simpler).

# Part 3: Implement Tables and ETL Procedure

## 3.1 Implementation using Microsoft SQL Server

The data warehouse was implemented in Microsoft SQL Server using SQL queries.

### Creation of Data Warehouse Tables

A SQL script was written to create the fact and dimension tables. As part of this step, both primary keys and foreign keys for each table were specified along with user-defined constraints for numerical attributes such as order quantity, gross total & discount.

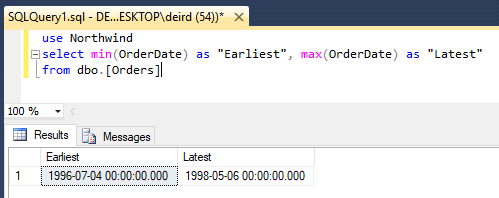
Please see [Appendix 1](#Appendix1) for the SQL scripts used.

### Populating the Data Warehouse using ETL Procedure

An ETL procedure was created to extract data from the operational database, transform it into the format required by the data warehouse and then load it into the data warehouse tables.

The ETL procedure extracts the data for the four “real” dimensions (Customer, Employee, Product and Supplier) from the operational database via a SQL SELECT query and then loads it into the data warehouse using the INSERT statement.

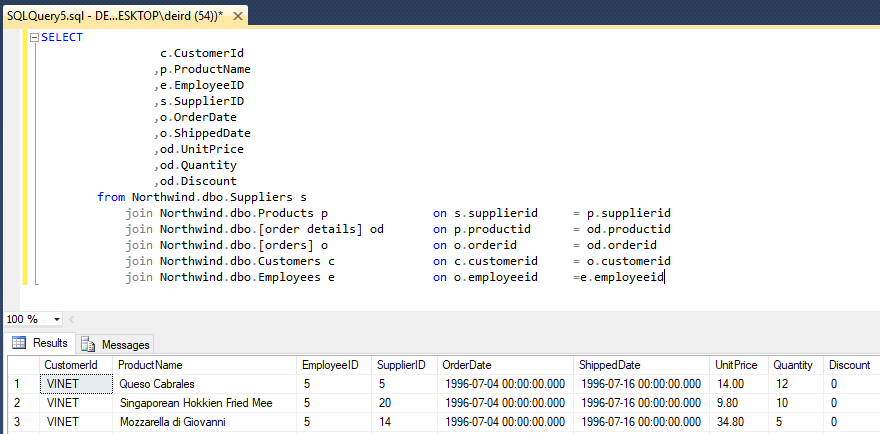
The Date dimension in the data warehouse is at the grain of individual days and represents the date each product was ordered. Before populating the Date dimension, a query was executed against the Orders table in the Northwind operational database to determine the date range required to be stored in the data warehouse. This query retrieved the earliest Order Date along with the latest Order Date from the Orders table.



**Fig 3**: Determining the Date Range required for the Date dimension

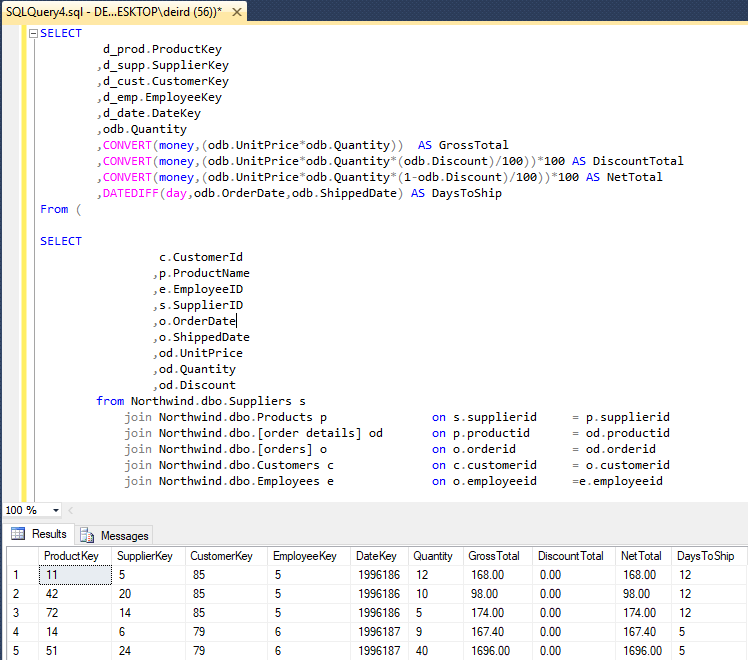
These dates were then used to set the start and end date variables which specify the scope of the SQL WHILE loop that populates the Date dimension.

To populate the fact table, it’s first necessary to extract the data required from the operational database. A subquery is used to perform a join across six of the tables in the operational database (Orders, Order Details, Products, Suppliers, Customers and Employees) and extract the operational keys for these tables along with the fields needed to calculate the derived facts (ie. UnitPrice, Quantity & Discount).



**Fig 4**: Extracting Data from the Operational Database

This subquery is then used inside a SELECT statement that transforms the data returned from the operational database into the format required by the fact table.



**Fig 5**: Transforming Data into the Format Required by the Data Warehouse

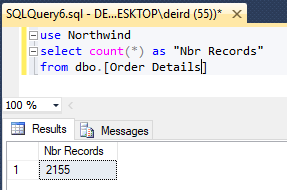
This transformation is where the ETL process matches the data extracted from the Northwind operational database with the dimension tables in the data warehouse and then creates the derived facts using SQL operators for subtraction, multiplication and division along with inbuilt functions such as CONVERT and DATEDIFF.

Lastly, the transformed data is loaded into the fact table in the data warehouse using the INSERT statement.

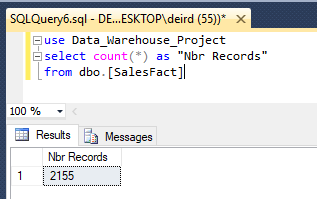
Please see [Appendix 2](#Appendix2) for the SQL scripts used.

### Validating the Data

Since the grain of the fact table in the data warehouse is at the same level of detail as the “Order Details” table in the Northwind operational database, a check was performed to ensure that the number of records in the fact table matched the number of records in the Order Details table.

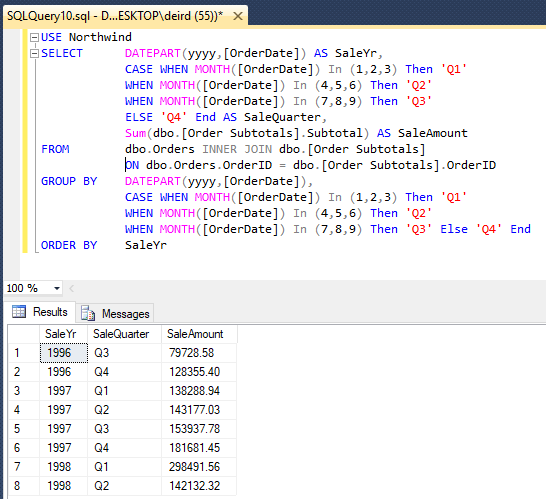


**Fig 6**: Retrieving the number of records from the Order Details table

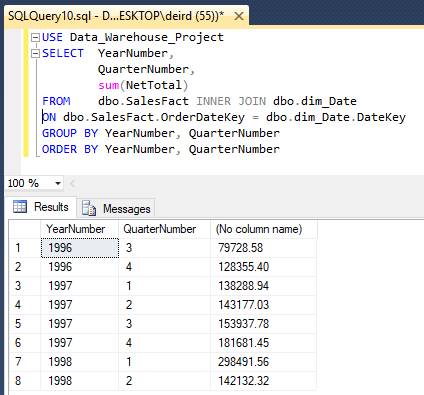


**Fig 7**: Retrieving the number of records from the Sales Fact table

A data validation check was also performed on the calculation of total net revenue per quarter in the data warehouse against the source data stored in the Northwind operational database. Results from the two queries executed proved that the calculations matched.



**Fig 8**: Aggregating net total per quarter in the Northwind operational database



**Fig 9**: Aggregating net total per quarter in the Northwind data warehouse

Please see [Appendix 3](#Appendix3) for the SQL scripts used.

## 3.2 Motivation for using Microsoft SQL Server

Microsoft SQL Server was chosen as the solution used to implement the data warehouse due mainly to availability of the software (I already had a copy of it installed on my laptop) as well as familiarity. Since I have used Microsoft SQL Server before and have experience writing SQL queries, there would be no learning curve involved. Hence, I felt that the task of implementing the data warehouse could be completed a lot quicker using Microsoft SQL Server than by opting to use a solution that would be new to me.

Another factor that contributed to the choice of SQL Server was the volume and variety of data involved. The Northwind operational database does not contain a large amount of data. In addition, the data is all structured (there is no unstructured or multimedia data involved). So, there was no requirement for a solution that could support Big Data.

## 3.3 Comparison of Microsoft SQL Server with Apache Spark

Apache Spark is a fast and general-purpose cluster computing system that is designed for large scale data processing. Spark is an open source software solution and is therefore free to download and use in production. Microsoft SQL Server is a relational database management system developed by Microsoft. SQL Server is proprietary software and comes with a high price tag.

SQL Server stores data tables by row, whereas Apache Spark stores data tables by column. Columnar databases boost query performance by reducing the amount of data that needs to be read from disk. For this reason, they are well-suited for [data warehouses](https://en.wikipedia.org/wiki/Data_warehouse) that involve highly complex queries which need to be executed over the entirety of data (potentially [petabytes](https://en.wikipedia.org/wiki/Petabyte) of data). Spark shines when there are huge volumes of data involved and is therefore a good choice for Big Data challenges.

Spark has a machine learning library and one of the features that sets it apart is its use of in-memory computation. For this reason, iterative applications are easily implemented in Spark, especially ones that involve machine learning or graph algorithms. SQL Server has only limited machine learning features beginning with the most recent versions (SQL Server 2016 & SQL Server 2017).

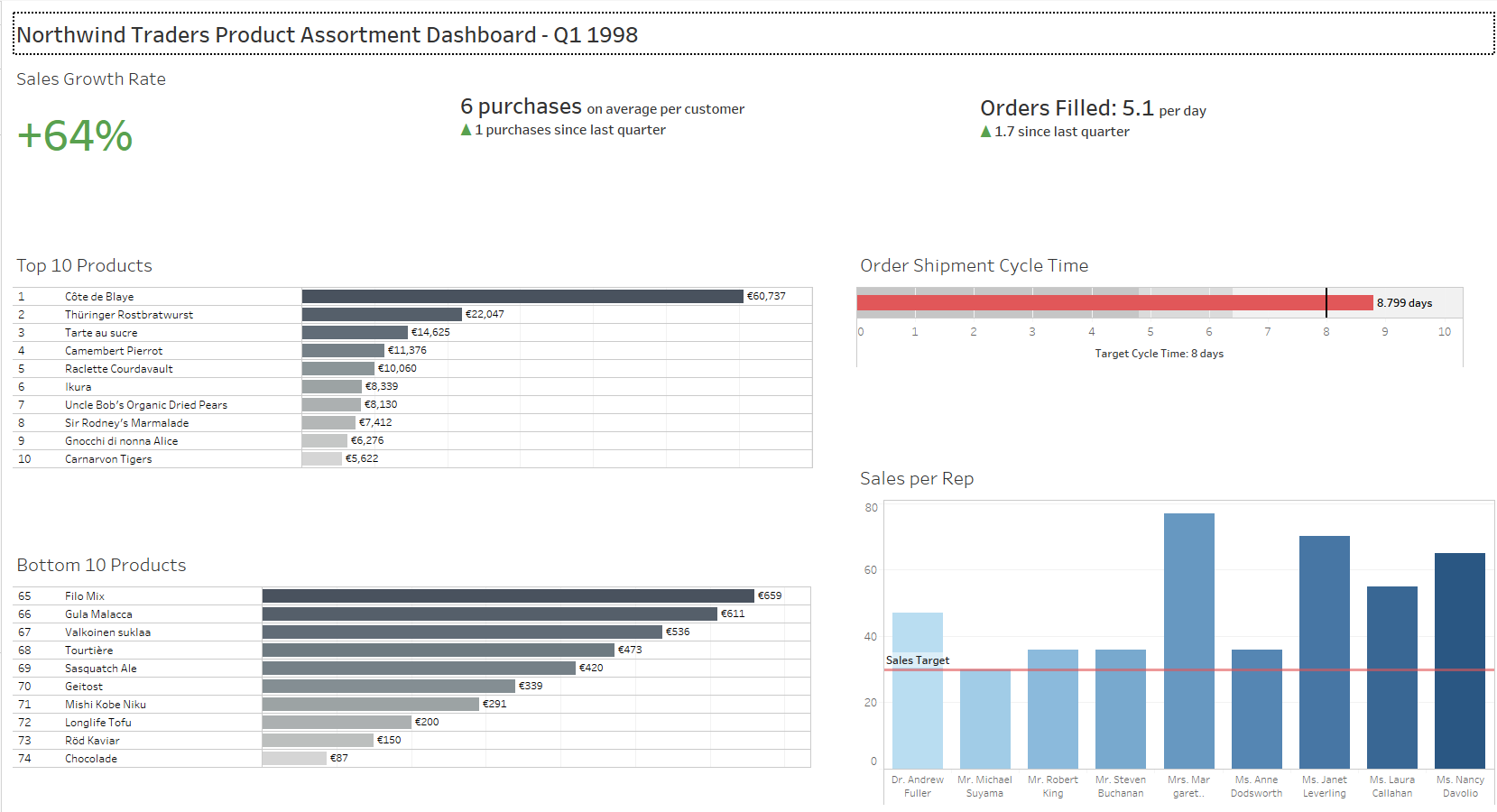
Spark is suitable for both batch-based processing and real-time processing. Whereas, SQL Server only supports batch-based processing.

Lastly, Spark supports both unstructured and multimedia data as well as traditional structured data. Whereas, SQL Server (being a relational database) only supports structured data.

# Part 4: Reporting and Analysis

## 4.1 Dashboard Design

A custom designed dashboard was developed in Tableau to measure the Key Performance Indicators defined in [Section 1.5](#KPIs) above.

**Fig 10**: Product Assortment Dashboard for Northwind Traders

This dashboard measures the Company’s performance over the previous quarter and displays all the required performance indicators on a single screen. The dashboard was developed following the six key design principles below:

* More whitespace is better
* Don’t make me think
* Don’t use a chart when a single number will do
* No more than 7 metrics
* Percentage differences complement tiles very well
* Allow scrolling or else declutter

Adhering to these design principles guarantees that information on the dashboard is displayed clearly and ensures that it can be digested by Business users in a single glance.

A minimal colour scheme has been employed in order to avoid distracting the user from the focus of the dashboard (i.e. the metrics).

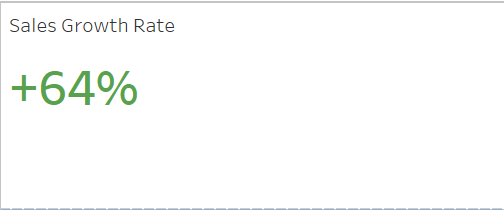
In keeping with the principle “No more than 7 metrics”, the dashboard consists of 3 tiles and 4 charts as follows;

|  |  |
| --- | --- |
| Dashboard Component | KPI |
| Tile 1 | Sales Growth Rate |
| Tile 2 | Purchase Frequency |
| Tile 3 | Sales Orders Filled per Day |
| Chart 1 | Top 10 Products |
| Chart 2 | Bottom 10 Products |
| Chart 3 | Order Shipment Cycle Time |
| Chart 4 | Sales per Rep |

**Table 2**: Dashboard Design

Each of the components on the dashboard is filtered by two dimensions; Year Number and Quarter Number (from the Date dimension table). This illustrates the benefit of having the Date dimension included in the data warehouse schema – it allows business users to filter by these time periods without having to create calculated fields or write SQL date functions in the BI tool.

### **Tile 1: Sales Growth Rate**

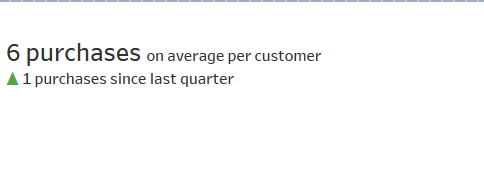


**Fig 11**: Tile displaying “Sales Growth Rate”

Since most users have been taught to read from left to right, the dashboard has been designed accordingly. For this reason, Sales Growth Rate has been placed in the top left corner. Since this KPI will monitored most frequently by Business users and their eyes will naturally be drawn to the top left corner when they first look at the dashboard, it will be easy for them to find this information.

Following the design principle “percentage differences complement tiles very well”, this KPI has been presented on a tile. A calculated field has been used to display a “+” sign if the Sales Growth Rate is positive and “-“ sign if it’s negative. In addition, the Sales Growth Rate is displayed in green font if it is positive and in red font if it is negative. This use of colour-coded font formatting will highlight exceptions that require action on the part of Senior Management.

### **Tile 2: Purchase Frequency**

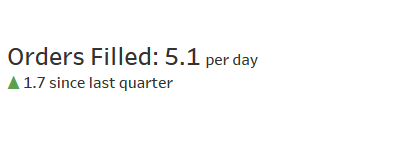


**Fig 12**: Tile displaying “Purchase Frequency”

Purchase Frequency measures the number of purchases on average per customer for the previous quarter. This KPI is presented on a tile because it is a single number and therefore does not require a chart. This follows the design principle “Don’t use a chart when a single number will do”.

A contextual metric has also been added to the tile to provide a comparison of this KPI against last quarter. The difference in purchase frequency since last quarter is displayed along with arrows indicating if the metric has increased or decreased. A calculated field has been used to display “▲” if the purchase frequency is up since last quarter and “▼” if the purchase frequency is down. Colour-coding of these arrows has been used to provide a visual cue to highlight exceptions that require action by Senior Management.

### **Tile 3: Sales Orders Filled per Day**



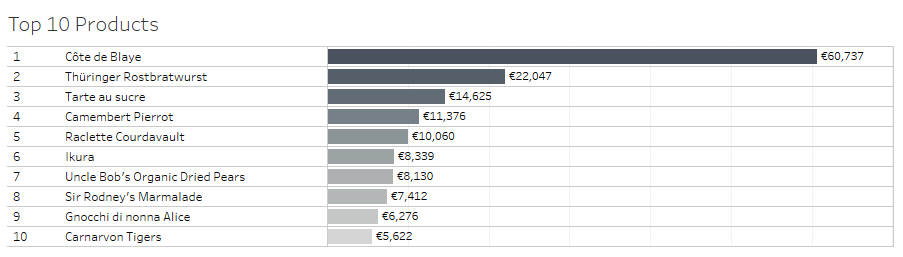
**Fig 13**: Tile displaying “Sales Orders Filled per Day”

Sales Order Filled per Day measures the average number of sales orders filled per day over the course of the previous quarter. Again, following the design principle “Don’t use a chart when a single number will do”, this metric has been displayed on a tile because it is a single number.

A contextual metric has also been added to this tile to provide a comparison of this KPI against last quarter. Colour-coded arrows indicate if the number of sales orders filled per day has increased or decreased since last quarter along with the actual difference. The colour-coding of the arrows again provides a visual cue to highlight any exceptions that require action by Senior Management.

Grouping the 3 tiles together at the top of the dashboard allows users to see how each area of the Business works together, since each of the metrics displayed on the tiles measures a different strategic objective of the Company (Sales Growth Rate measures one of the Financial objectives, Purchase Frequency measures one of the Customer objectives and Sales Orders Filled per Day measures one the Internal Business Process objectives). Placing this unrelated information together, while seemingly counter-intuitive, may actually lead to insights about the Company’s performance – for example, if an increase in sales growth rate is perhaps related to an increase in purchase frequency, increased efficiency in the processing of sales orders or another seemingly unrelated factor. This design also allows Senior Management to maintain a “big picture” view when looking at the Company’s performance.

### **Chart 1: Top 10 Products**



**Fig 14**: Chart displaying “Top 10 Products”

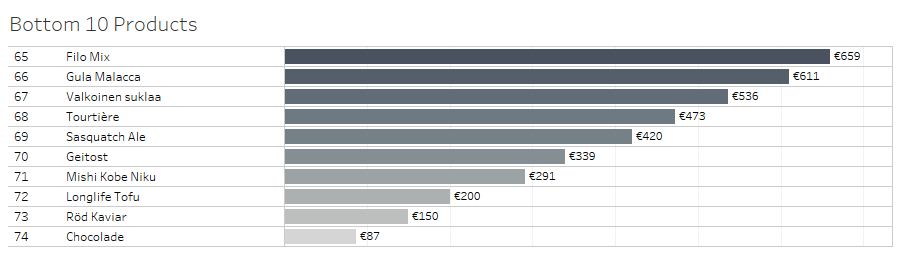
The Top 10 Products chart ranks products based on revenue performance and informs Business users which products are selling well.

A horizontal bar chart has been used to display the ranking for the product along with its name and the total revenue generated by the product over the previous quarter. Colour-coding has been used to display the bar marks for higher revenue values in a darker colour. This allows business users to quickly and easily consume the information and compare the top performing products.

The chart has been filtered to show performance for the top 10 products only. Since there were 74 different products with sales orders over the course of the previous quarter, it would not be feasible to show the performance for all products. In addition, displaying all products would clutter up the dashboard. So, following the design principle “Allow scrolling or else declutter”, the chart is filtered to show performance for the top 10 products only.

The quantity sold has been added to the chart as a tooltip, so that Business users can also view the volume of each product sold.

### **Chart 2: Bottom 10 Products**



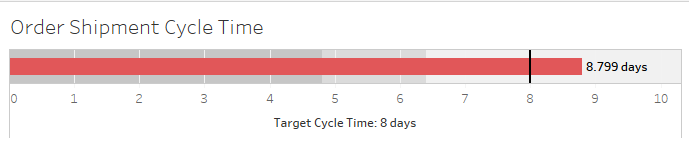
**Fig 15**: Chart displaying “Bottom 10 Products”

The Bottom 10 Products chart ranks the poorest performing products in terms of revenue and identifies products that are failing to resonate with customers. Since the focus of analysis in the dashboard is product assortment, this bar chart plays a key role in telling the “story” of the data.

For consistency, the Bottom 10 Products are displayed using a horizontal bar chart in exactly the same manner as the Top 10 Products. The ranking for each product is displayed along with its name and the total revenue generated by the product over the previous quarter. Colour-coding has again been used to display the bar marks for higher revenue values in a darker colour. This allows business users to quickly and easily compare the poorly performing products. The quantity sold has also been added to the chart as a tooltip, so that Business users can view the volume of each product sold.

This bar chart has again been filtered (in this case, to show the lowest 10 performing products) in order to avoid cluttering up the dashboard.

### **Chart 3: Order Shipment Cycle Time**



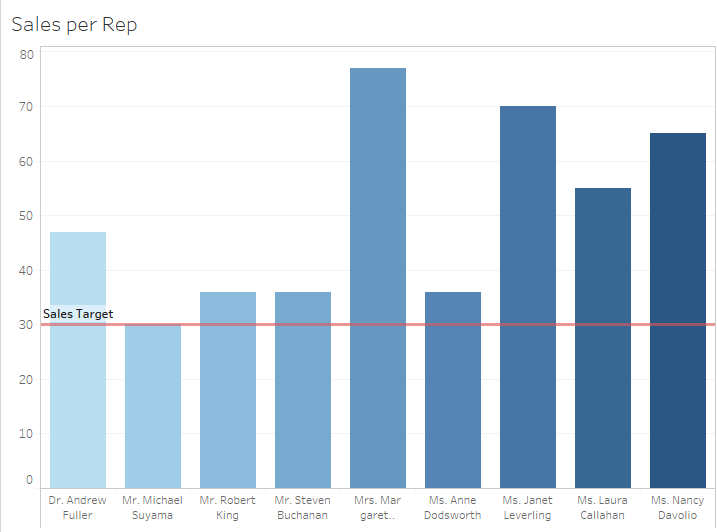
**Fig 16**: Bullet Graph displaying “Order Shipment Cycle Time”

Order Shipment Cycle Time measures the average number of days between the order being placed and the shipment of the order.

This KPI is benchmarked against the industry standard. For this reason, a bullet graph has been chosen, since it is an effective means of displaying a single measure (ie. average cycle time) along with a contextual metric (ie. target cycle time) to provide a comparative measure.

Presenting the information in a bullet graph allows Business users to rapidly monitor performance, since they can perceive at a glance if the target cycle time has been hit or not. This follows the design principle “Don’t make me think”. In addition, bullet graphs also don’t take up much screen space and therefore don’t clutter up the dashboard. This follows the design principle “More whitespace is better”.

### **Chart 4: Sales per Rep**



**Fig 17**: Bar chart displaying “Sales per Rep”

Sales per Rep measures the performance of each of the sales reps over the course of the previous quarter. Performance for the sales reps is also benchmarked against a pre-defined sales target (30 sales per quarter).

Since in this case, there are only nine sales reps, it makes sense to display this information using a bar chart. This allows Senior Management to quickly and easily compare the performance of each of the sales reps. The addition of a reference line at the target mark also allows Senior Management to instantly identify which employees are meeting expected sales targets and which aren’t. Presenting the information in this way follows the design principle “Don’t make me think”.

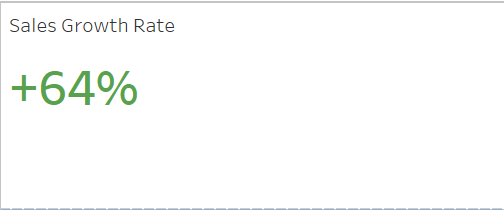
The Hire Date of the sales rep has also been added to the chart as a tooltip, so that Business users can easily identify experienced sales reps versus newly joined reps (this is useful information to have on hand, as it may help to explain poor performance in some cases).

## 4.2 Dashboard Analysis

For the purposes of analysis, Q1 1998 has been selected as the reporting period of interest.

### **KPI 1: Sales Growth Rate**

This KPI was chosen because it provides a measure of the pace at which the organization's sales revenue is increasing or decreasing. This allows Senior Management to monitor the Company's progress against the strategic objective of increasing sales revenue.

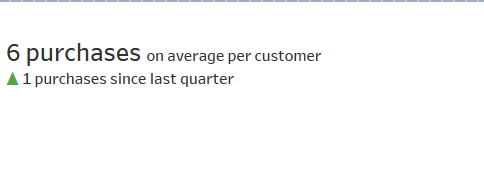


**Fig 18**: Sales Growth Rate for Q1 1998

Quarter 1 of 1998 was a particularly strong quarter for the Company and its sales growth rate compared to Q4 1997 was +64%. This was a significant increase compared to previous quarters, since the sales growth rate in Q4 1997 was just 18%, in Q3 1997 it was 7.5% and in Q1 1997 it was only 3.5%. This metric tells us that the Company more than tripled its sales growth rate in Q1 1998 and in fact, Q1 1998 is the strongest quarter to date for the Company in terms of sales revenue.

### **KPI 2: Purchase Frequency**

This KPI was chosen because it provides insight into the appeal of products. This metric helps Senior Management to monitor the Company's progress against the strategic objective of increasing product appeal.

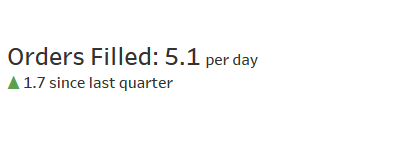


**Fig 20**: Purchase Frequency for Q1 1998

In Q1 1998, 6 purchases on average were made per customer (which was up 1 since Q4 1997). This is the highest purchase frequency so far, since in previous quarters, the purchase frequency was either 4 or 5 purchases per customer on average.

### **KPI 3: Sales Orders Filled per Day**

This KPI was chosen because it provides insight into how quickly sales orders are being filled. This metric helps Senior Management to monitor the Company's progress against the strategic objective of filling sales orders on time.

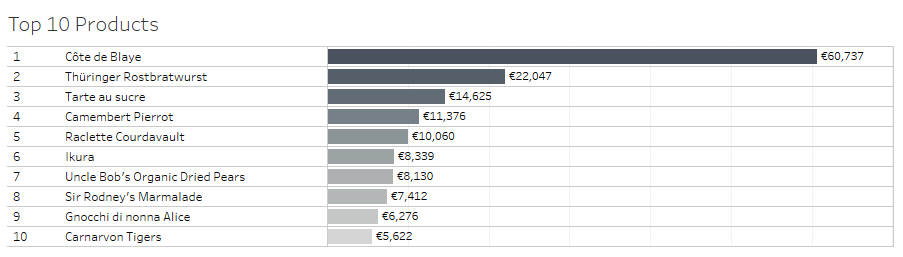


**Fig 22**: Sales Orders Filled per Day in Q1 1998

In Q1 1998, 5.1 sales orders were filled per day on average (which was up 1.7 since the previous quarter). This is considerably higher than previous quarters which had an average number of sales orders filled per day of around 2.7/2.8. Prior to Q1 1998, the previous highpoint was 3.4 sales orders filled per day. More orders were filled per day in Q1 1998 as a result of the large increase in sales orders received (452 in total compared to 309 in Q4 1997).

### **KPI 4: Top 10 Products**

This KPI was chosen because it informs the sales team which products are selling well. This metric helps Senior Management to monitor the Company's progress against the strategic objective of maximizing product sales.



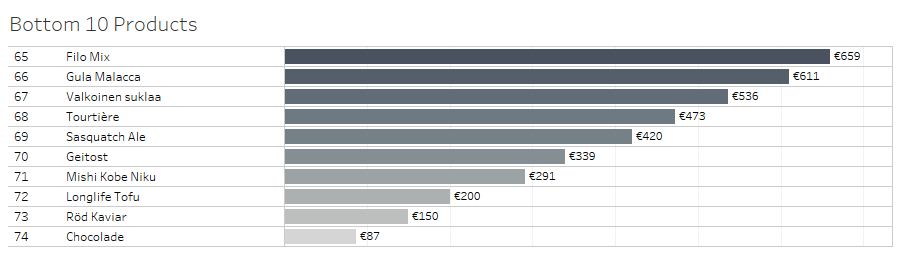
**Fig 24**: Top 10 Products for Q1 1998

In Q1 1998, Côte de Blaye was by far the top selling product, bringing in revenue of over €60k for the Company. Interestingly, this product did not feature in the top 10 products for the previous quarter (Q4 1997). And in Q3 1997, it brought in only €7.3k revenue. However, in Q1 1997, it was also the top selling product, bringing in €25k revenue for the Company.

It would appear therefore that this product is some type of high-end wine or champagne, since it’s listed in the “Beverages” category. The strong performance in Q1 each year is likely due to seasonality with increased sales over the Christmas & New Year period, although it’s also possible that the Company runs a promotion on this product at this particular time each year which helps to drive up the sales.

### **KPI 5: Bottom 10 Products**

This KPI was chosen because it informs the sales team which products are failing to resonate with customers and may require special attention. This metric also allows Senior Management to make informed decisions as to whether certain products should be discontinued.



**Fig 26**: Bottom 10 Products for Q1 1998

In Q1 1998, “Chocolade” was the poorest performing product, bringing in only €87 in revenue for the Company.

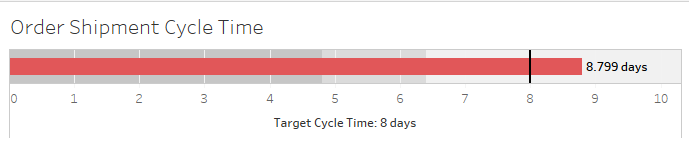
Looking back over previous quarters, this product has featured in the Bottom 10 Products for every quarter since Q2 1997. The Company may therefore wish to consider discontinuing this product, so that it can be replaced with a product that will be of greater appeal to its customers.

Several other products have twice featured in the Bottom 10 Products over the past four quarters and the Company may wish to keep a close eye on the performance of these going forward:

* Filo Mix
* Laughing Lumberjack Lager
* Chef Anton's Gumbo Mix
* Guarana Fantastica
* Escargots de Bourgogne
* Konbu

### **KPI 6: Order Shipment Cycle Time**

This KPI was chosen because it provides insight into the efficiency of the supply chain process. This metric helps Senior Management to monitor the Company's progress against the strategic objective of moving products quickly.



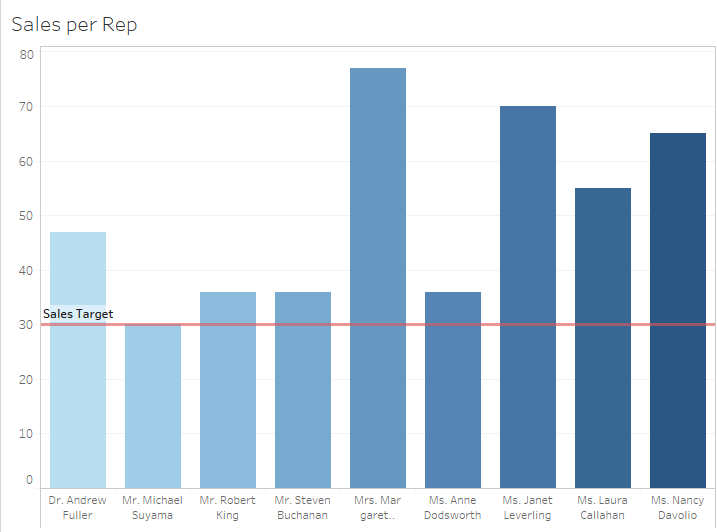
**Fig 27**: Order Shipment Cycle Time for Q1 1998

In Q1 1998, the order shipment cycle time was 8.799 days which exceeded the target cycle time of 8 days. The fact that the target cycle time wasn’t hit in Q1 1998 means that the Company didn’t move products as quickly as it would have liked. The supply chain process may have been under increased strain as a result of the large increase in sales orders that were received in Q1 1998.

Senior Management may decide to monitor the sales growth rate on a more frequent basis going forward (e.g. perhaps monthly), so that it can put extra resources in place to handle the shipment of orders when there is a spike in the number of sales orders received. This will help maintain efficiency in the supply chain process and guarantee that orders are shipped to customers in a timely manner.

### **KPI 7: Sales per Rep**

This KPI was chosen because it measures the ability of each of the sales reps to generate revenue for the organization. This metric helps Senior Management to monitor the Company's progress against the strategic objective of improving employee focus on sales.



**Fig 28**: Sales per Rep in Q1 1998

In Q1 1998, all of the sales reps hit the expected sales target of 30 sales per quarter. Margaret Peacock performed best, converting 77 sales for the Company over the course of the quarter (which was over double the sales target). Janet Leverling and Nancy Davolio were the next best performers, both achieving over 65 sales each.

In previous quarters, Anne Dodsworth and Steven Buchanan were the poorest performing sales reps. However, the fact that they hit the sales target in Q1 1998 indicates that the Company’s strategy of improving employee focus on sales appears to be working.

# Appendix 1 – SQL Script to Build the Data Warehouse

|  |
| --- |
| --Create Customer Dimensions  CREATE TABLE dbo.dim\_Customer (  CustomerKey INT IDENTITY(1, 1) PRIMARY KEY,  CustomerID NCHAR(5) NOT NULL,  CustomerName NVARCHAR(40) NOT NULL,  CustomerCity NVARCHAR(15) NULL,  CustomerRegion NVARCHAR(15) NULL,  CustomerCountry NVARCHAR(15) NULL  )  --Create Product Dimensions  CREATE TABLE dbo.dim\_Product (  ProductKey INT IDENTITY(1, 1) PRIMARY KEY,  ProductID INT NOT NULL,  ProductName NVARCHAR(40) NOT NULL,  CategoryName NVARCHAR(15) NOT NULL,  QuantityPerUnit NVARCHAR(20) NULL,  UnitPrice SMALLMONEY NULL CONSTRAINT "Products\_UnitPrice" DEFAULT (0),  UnitsInStock SMALLINT NULL CONSTRAINT "Products\_UnitsInStock" DEFAULT (0),  Discontinued BIT NOT NULL  )  --Create Supplier Dimensions  CREATE TABLE dbo.dim\_Supplier(  SupplierKey INT IDENTITY(1, 1) PRIMARY KEY,  SupplierID INT NOT NULL,  SupplierName NVARCHAR(40) NOT NULL,  SupplierCity NVARCHAR (15) NULL,  SupplierRegion NVARCHAR(15) NULL,  SupplierCountry NVARCHAR(15) NULL  )  --Create Employee Dimensions  CREATE TABLE dbo.dim\_Employee (  EmployeeKey INT IDENTITY(1, 1) PRIMARY KEY,  EmployeeID INT NOT NULL,  EmployeeName NVARCHAR(85) NOT NULL,  Title NVARCHAR(30) NULL,  HireDate DATETIME NULL ,  EmployeeCity NVARCHAR(15) NULL,  EmployeeRegion NVARCHAR(15) NULL,  EmployeeCountry NVARCHAR(15) NULL  )  --Create Date Dimensions  CREATE TABLE dbo.dim\_Date (  DateKey INT NOT NULL PRIMARY KEY,  CalendarDate DATE NOT NULL,  FullDateDesc VARCHAR(60) NOT NULL,  DayNumber INT NOT NULL,  WeekdayName VARCHAR(15) NOT NULL,  WeekNumber INT NOT NULL,  CalMonthNumber INT NOT NULL,  CalMonthName VARCHAR (15) NOT NULL,  QuarterNumber INT NOT NULL,  YearNumber INT NOT NULL  )  --Create Sales Facts  CREATE TABLE SalesFact (  FactID INT IDENTITY(1, 1) PRIMARY KEY,  ProductKey INT NOT NULL,  SupplierKey INT NOT NULL,  CustomerKey INT NOT NULL,  EmployeeKey INT NOT NULL,  OrderDateKey INT NOT NULL,  OrderQuantity SMALLINT NOT NULL CONSTRAINT "SalesFact\_Quantity" DEFAULT (1),  GrossTotal SMALLMONEY NOT NULL CONSTRAINT "SalesFact\_GrossTotal" DEFAULT (0),  DiscountTotal SMALLMONEY NOT NULL CONSTRAINT "SalesFact\_Discount" DEFAULT (0),  NetTotal SMALLMONEY NOT NULL CONSTRAINT "SalesFact\_NetTotal" DEFAULT (0),  DaysToShip INT NULL,  FOREIGN KEY (ProductKey) REFERENCES dim\_Product(ProductKey),  FOREIGN KEY (SupplierKey) REFERENCES dim\_Supplier(SupplierKey),  FOREIGN KEY (CustomerKey) REFERENCES dim\_Customer(CustomerKey),  FOREIGN KEY (EmployeeKey) REFERENCES dim\_Employee(EmployeeKey),  FOREIGN KEY (OrderDateKey) REFERENCES dim\_Date(DateKey),  ) |

# Appendix 2 - ETL Script to Populate the Data Warehouse

|  |
| --- |
| ---------------------------------------------------------------------------------------  -- POPULATE CUSTOMER DIMENSIONS --  ---------------------------------------------------------------------------------------  INSERT dim\_Customer ( CustomerID, CustomerName, CustomerCity, CustomerRegion, CustomerCountry)  SELECT  customerid,  companyname,  city,  region,  country  FROM Northwind.dbo.Customers  ---------------------------------------------------------------------------------------  -- POPULATE EMPLOYEE DIMENSIONS --  ---------------------------------------------------------------------------------------  INSERT dim\_Employee (EmployeeID, EmployeeName, Title, HireDate, EmployeeCity, EmployeeRegion, EmployeeCountry)  SELECT  EmployeeID,  TitleOfCourtesy + ' ' + FirstName + ' ' + LastName AS EmployeeName,  title,  HireDate,  city,  region,  country  FROM Northwind.dbo.Employees  ---------------------------------------------------------------------------------------  -- POPULATE PRODUCT DIMENSIONS --  ---------------------------------------------------------------------------------------  INSERT dim\_Product (ProductID, ProductName, CategoryName, QuantityPerUnit, UnitPrice, UnitsInStock, Discontinued)  SELECT  ProductID,  ProductName,  CategoryName,  QuantityPerUnit,  UnitPrice,  UnitsInStock,  discontinued  FROM Northwind.dbo.Products  INNER JOIN Northwind.dbo.Categories  ON Northwind.dbo.Products.CategoryID = Northwind.dbo.Categories.CategoryID  ---------------------------------------------------------------------------------------  -- POPULATE SUPPLIER DIMENSIONS --  ---------------------------------------------------------------------------------------  INSERT dim\_Supplier (SupplierID, SupplierName, SupplierCity, SupplierRegion, SupplierCountry)  SELECT  supplierid,  companyname,  city,  region,  country  FROM Northwind.dbo.Suppliers  ---------------------------------------------------------------------------------------  -- POPULATE DATE DIMENSIONS --  ---------------------------------------------------------------------------------------  DECLARE  @start\_date\_dt SMALLDATETIME,  @end\_date\_dt SMALLDATETIME,  @sql\_string NVARCHAR(1024),  @DateKey INT,  @calendar\_date\_dt DATE,  @FullDateDesc VARCHAR(60),  @WeekDayName VARCHAR(10),  @DayNumber INT,  @calendar\_day\_of\_year\_num INT,  @WeekNumber INT,  @CalMonthNumber INT,  @CalMonthName VARCHAR(10),  @QuarterNumber INT,  @YearNumber INT  SET @calendar\_date\_dt = '1996-07-04'  SET @end\_date\_dt = '1998-06-11'  WHILE (@calendar\_date\_dt <= @end\_date\_dt)  BEGIN  SELECT  @WeekDayName = DATENAME(WEEKDAY, @calendar\_date\_dt),  @DayNumber = DATEPART(DD, @calendar\_date\_dt),  @calendar\_day\_of\_year\_num = DATEPART(DY, @calendar\_date\_dt),  @WeekNumber = DATEPART(WK, @calendar\_date\_dt),  @CalMonthNumber = DATEPART(M, @calendar\_date\_dt),  @CalMonthName = DATENAME(MONTH, @calendar\_date\_dt),  @QuarterNumber = DATEPART(QQ, @calendar\_date\_dt),  @YearNumber = DATEPART(YYYY, @calendar\_date\_dt),  @DateKey = CAST(CAST(@YearNumber AS VARCHAR) + RIGHT('00' + CAST(@calendar\_day\_of\_year\_num AS VARCHAR), 3) AS INT),  @FullDateDesc = @CalMonthName + ' ' + CAST(@DayNumber AS VARCHAR) + ', ' + CAST(@YearNumber AS VARCHAR)    SELECT @sql\_string =  'INSERT INTO dbo.dim\_Date' +  ' (' +  'DateKey, ' +  'CalendarDate, ' +  'FullDateDesc, ' +  'DayNumber,' +  'WeekDayName,' +  'WeekNumber,' +  'CalMonthNumber,' +  'CalMonthName,' +  'QuarterNumber, ' +  'YearNumber' +  ') ' +  'VALUES ' +  '(' +  CHAR(39) + CAST(@DateKey AS VARCHAR) + CHAR(39) + ',' +  CHAR(39) + CAST(@calendar\_date\_dt AS VARCHAR) + CHAR(39) + ',' +  CHAR(39) + @FullDateDesc + CHAR(39) + ',' +  CAST(@DayNumber AS VARCHAR) + ',' +  CHAR(39) + @WeekDayName + CHAR(39) + ',' +  CAST(@WeekNumber AS VARCHAR) + ',' +  CAST(@CalMonthNumber AS VARCHAR) + ',' +  CHAR(39) + @CalMonthName + CHAR(39) + ',' +  CAST  (@QuarterNumber AS VARCHAR) + ',' +  CAST(@YearNumber AS VARCHAR) + ')'  EXEC sp\_executesql @sql\_string  SET @calendar\_date\_dt = DATEADD(day,1,@calendar\_date\_dt)  END  ---------------------------------------------------------------------------------------  -- POPULATE SALES FACT --  ---------------------------------------------------------------------------------------  INSERT INTO [dbo].[SalesFact]  (ProductKey  ,SupplierKey  ,CustomerKey  ,EmployeeKey  ,OrderDateKey  ,OrderQuantity  ,GrossTotal  ,DiscountTotal  ,NetTotal  ,DaysToShip  )  SELECT  d\_prod.ProductKey  ,d\_supp.SupplierKey  ,d\_cust.CustomerKey  ,d\_emp.EmployeeKey  ,d\_date.DateKey  ,odb.Quantity  ,CONVERT(money,(odb.UnitPrice\*odb.Quantity)) AS GrossTotal  ,CONVERT(money,(odb.UnitPrice\*odb.Quantity\*(odb.Discount)/100))\*100 AS DiscountTotal  ,CONVERT(money,(odb.UnitPrice\*odb.Quantity\*(1-odb.Discount)/100))\*100 AS NetTotal  ,DATEDIFF(day,odb.OrderDate,odb.ShippedDate) AS DaysToShip  From (  SELECT  c.CustomerId  ,p.ProductName  ,e.EmployeeID  ,s.SupplierID  ,o.OrderDate  ,o.ShippedDate  ,od.UnitPrice  ,od.Quantity  ,od.Discount  from Northwind.dbo.Suppliers s  join Northwind.dbo.Products p on s.supplierid = p.supplierid join Northwind.dbo.[order details] od on p.productid = od.productid  join Northwind.dbo.[orders] o on o.orderid = od.orderid  join Northwind.dbo.Customers c on c.customerid = o.customerid  join Northwind.dbo.Employees e on o.employeeid =e.employeeid  )  as odb  join dim\_Customer d\_cust on d\_cust.CustomerID = odb.CustomerId  join dim\_Product d\_prod on d\_prod.ProductName = odb.ProductName  join dim\_Employee d\_emp on d\_emp.EmployeeID = odb.EmployeeID  join dim\_Supplier d\_supp on d\_supp.SupplierID = odb.SupplierID  join dim\_Date d\_date on d\_date.CalendarDate = odb.OrderDate |

# Appendix 3 - ETL Data Validation Scripts

|  |
| --- |
| ---------------------------------------------------------------------------------------  -- Count Number of Records in Order Details table  ---------------------------------------------------------------------------------------  USE Northwind  SELECT count(\*) as "Nbr Records"  FROM dbo.[Order Details]  ---------------------------------------------------------------------------------------  -- Count Number of Records in Sales Fact table  ---------------------------------------------------------------------------------------  USE Data\_Warehouse\_Project  SELECT count(\*) as "Nbr Records"  FROM dbo.[SalesFact]  ---------------------------------------------------------------------------------------  -- AGGREGATE NET TOTAL BY QUARTER IN NORTHWIND OPERATIONAL DB --  ---------------------------------------------------------------------------------------  USE Northwind  SELECT DATEPART(yyyy,[OrderDate]) AS SaleYr,  CASE WHEN MONTH([OrderDate]) In (1,2,3) Then 'Q1'  WHEN MONTH([OrderDate]) In (4,5,6) Then 'Q2'  WHEN MONTH([OrderDate]) In (7,8,9) Then 'Q3'  ELSE 'Q4' End AS SaleQuarter,  Sum(dbo.[Order Subtotals].Subtotal) AS SaleAmount  FROM dbo.Orders INNER JOIN dbo.[Order Subtotals] ON dbo.Orders.OrderID = dbo.[Order Subtotals].OrderID  GROUP BY DATEPART(yyyy,[OrderDate]),  CASE WHEN MONTH([OrderDate]) In (1,2,3) Then 'Q1'  WHEN MONTH([OrderDate]) In (4,5,6) Then 'Q2'  WHEN MONTH([OrderDate]) In (7,8,9) Then 'Q3'  ELSE 'Q4' End  ORDER BY SaleYr  ---------------------------------------------------------------------------------------  -- AGGREGATE NET TOTAL BY QUARTER IN NORTHWIND DATA WAREHOUSE --  ---------------------------------------------------------------------------------------  USE Data\_Warehouse\_Project  SELECT YearNumber,  QuarterNumber,  sum(NetTotal)  FROM dbo.SalesFact INNER JOIN dbo.dim\_Date ON dbo.SalesFact.OrderDateKey = dbo.dim\_Date.DateKey  GROUP BY YearNumber, QuarterNumber  ORDER BY YearNumber, QuarterNumber |

# References

* Product Assortment: Definition & Strategy:

<http://study.com/academy/lesson/product-assortment-definition-strategy-quiz.html>

(Accessed: 29/10/2017)

* Sales Metrics and KPIs:

<https://www.klipfolio.com/resources/kpi-examples/sales>

(Accessed: 29/10/2017)

* The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modelling

Kimball, R. & Moss, M.

John Wiley & Sons Inc. (2013)

* Dashboard Design Tips

<https://docs.google.com/presentation/d/1weOIiW02Fm_Tv--e3P9ZEpbBwLS3Fs1Az9HYMoFggxM/edit#slide=id.g29795cb2e8_0_60>

(Accessed: 2/12/2017)

* 3 Simple Rules for Intuitive Dashboard Design:

<https://www.klipfolio.com/blog/intuitive-dashboard-design>

(Accessed: 9/12/2017)