IN-MEMORY DATA MODELLING, ETL AND ANALYSIS

Report submission as a requirement for the module of "Data Warehousing"

MSc Degree in Data Analysis
at the school of computing
Robert Gordon University
Aberdeen, Scotland
November 2020

Table of Contents

Introduction

In this project, I will identify and explain the main concepts of a data warehouse. I will also describe, analyse and apply methodologies for designing data warehouse applications and OLAP.

Firstly, I will explain the project specifications; then I will move to Data modelling where I aim to apply normalisation techniques. I will justify and document each table, including their relations, attributes and hierarchies. Once the tables are ready, I will implement a star schema and deploy my solution through Excel.

The second section of this project will be the testing of different calculated columns and measurements on the design to imitate a real case scenario and determine if the design is capable enough to be recommended for a business intelligence solution.

Project overview

A **superstore** with branches across North-America keeps a record of orders placed by customers in a single **Excel** spreadsheet (Coursework.xlsx). They recognise that this is not ideal and so they want you to organise the data in a way that better supports decisions.

Task 1

Using **Power Query** and **PowerPivot**, reorganise the data into an in-memory normalised schema that allows you to get rid of data redundancy. Justify your data modelling decisions, including

- Creation of new tables and whether they are fact or dimension.
- why attributes belong to a particular table.
- the relationship between tables.
- Any hierarchies built in the tables.

The superstore is interested in analysing **transactions** by **Date** (Year, Month, Quarter, Day), **Product**, **Customer**, **Store**, and **Region**.
Although the records held in the spreadsheet span the years 1997 and 1998, your solution should accommodate records being available for later years. Feel free to test this by adding spurious data of your own.

Task 2

To support the superstore with their data analysis, you will have to create calculated columns or measures in the appropriate tables using **PowerPivot** and **DAX** formulas. You will also have to answer questions related to their calculated columns/measures as well as create and display some **Pivot Charts.**

Data Modelling and ETL

The table provided for this project (Coursework.xlsx) has 269,721 rows and 47 columns. The goal is to model this table using ETL techniques in Excel to create a database fit for the task of business analysis and decision support towards the superstore.

Normalisation

I will be basing my normalisation criteria on **OLAP** which focuses on complex and analytical queries of existing data instead of changing it; This means standard **OLTP** normalisation will not be the right approach because it partitions the data into tables and quiring them would require a high number of joins.
The normalisation technique I have applied is suited for **data warehouse** design based in a multidimensional model called **data cube**. This normalisation process allows a certain level of redundancy which is expected in these types of structures. I have considered each of the different dimension of the cube to be **business perspectives**. The superstore is interested in the analysis of transactions based on the following business perspectives (Date, product, store, customer) hence my criteria for dividing the table into 5 different dimensions/tables.

The following sections will present a set of 3 tables for each of the dimensions previously mentioned. These tables represent the **metadata**; this information will aim to describe each table as well as its attributes, relations and hierarchies.

Customer dimension table

This table looks at the business from the customer perspective; It can be used to study the different groups of customers, find trends or products they have in common and perhaps target products or advertisements to those groups.

- ✓ Table Name: **customer**
- ✓ Number of rows: **8842**
- ✓ Fact or Dimension: **Dimension**

Columns	Justification
customer_id	Uniquely identifies an instance in the customer entity.
customer_acct_num	Customers can own accounts for the superstore, this attribute allows to uniquely identify their accounts.
fisrt_name	A customer is a person, and a person has a first name. This attribute is a reference to theirs.
last_name	A customer is a person, and a person has a last name. This attribute is a reference to theirs.
customer_address	A customer has a home, this attribute is a reference their home's address.
customer_city	A customer home is within a city, this attribute refers to that city.
customer_state_province	A customer home belongs to a state or province, this attribute stores that province or state.
customer_postal_code	A customer home has a postal code, this attribute stores that code.
customer_country	A customer lives in a country. This attribute stores the country the customer lives in.
birthdate	Every customer has a birthdate. This attribute stores a customer's birthdate
marital_status	A customer can be married, single etc. This attribute stores their marital status.

yearly_income	A customer might have a job. This attribute stores their job's yearly income.
gender	A customer identifies himself as a man, women etc... This attribute stores the customer's gender.
total_children	A customer might have children. This attribute stores how many children they have.
num_children_at_home	A customer might have children that still live with them at their homes. This attribute stores how many still live with them at their homes.
education	A customer might have education or not. This attribute stores their level of education.
acct_open_date	A customer can register an account for the superstore. This attribute stores the date the account was created.
member_card	A customer can subscribe to a store membership. This attribute stores the membership category the customer is subscribed to.
occupation	A customer might have a job and a job falls into different categories. This attribute stores the customer's job category.
homeowner	A customer might own a home. This attribute stores weather the user owns a home or not.

Table 1: Metadata of Customer Table

Customer **relations**

The relation of customer towards transactions is **one to many**.

Table Name	Primary Key	Table	Foreign Key
customer	customer_id	transactions	customer_id

Table 2: Entity relation of Customer table

Customer **hierarchies**

- ✓ Table Name: **customer**
- ✓ Hierarchy Name: **Customer Location**

Attributes in Hierarchy	Justification
❖ Customer_country ❖ Customer_state_province ❖ Customer_city ❖ Customer_postal_code	This hierarchy allows us to drill down to a granularity level where we look at the business from the perspective of the customer location. The hierarchy makes sense as the relation parent and child is one to many.

Table 3: Customer hierarchies

Store dimension table

This table contains information of each store; it looks at the business from the store perspective. It can be used to study the store characteristic or location and decide if they are impacting the business.

- ✓ Table Name: **store**
- ✓ Number of rows: **24**
- ✓ Fact or Dimension: **Dimension**

Columns	Justification
store_id	Uniquely identifies a row in the store table.
store_name	Every store has a unique name to help representing them, this attribute stores such name.
store_type	Each store belongs to a certain category depending on its size (e.g. supermarket, grocery store), this attribute indicates which type of store it is.
store_street_address	Each store is located on a unique address, this attribute stores that address.
store_city	Each store belongs resides within a city, this attribute stores that city.
store_sales_district	A store might belong to a district, this attribute stores the district the store belongs to.
store_state	Each store is located within a state, this attribute stores that state.
store_region	Stores belong to different regions; this attribute stores the region a store belongs to.
store_country	Each store is located within a country, this attribute stores that country.
region_id	Similar regions might appear between different countries (e.g. west, east, central) this ID uniquely identifies the region and country. It is also faster to query than strings.

store_phone	Each store has a phone number as a point of contact, this attribute stores that phone number.
stock_date	Stores restock products on certain dates, this attribute keeps track of the dates the store re-stocked.
fisrt_opened_date	This attribute stores the opening date of a store.
last_remodel_date	This attribute stores the last date a store was remodelled.
total_sqft	This attribute stores the size of the store including parking and other areas in square foot.
grocery_sqft	his attribute stores the inside the store size in square foot.

Table 4: Metadata of Store Table

store **relations**

The relation of store towards transactions is **one to many**.

Table Name	Primary Key	Table	Foreign Key
store	store_id	transactions	store_id

Table 5: Entity relation of Store table

store **hierarchies**

- ✓ Table Name: **customer**
- ✓ Hierarchy Name: **Customer Location**

Attributes in Hierarchy	Justification
❖ store_country ❖ store_region ❖ store_state ❖ store_sales_district ❖ store_city	This hierarchy allows us to drill down to a granularity level where we look at the business from the perspective of the store locations. The hierarchy makes sense as the relation parent and child are one to many. With the exception of district which makes this a **ragged** hierarchy.

❖ store_type ❖ store_name	This hierarchy allows us to see the business from the type of store perspective allowing us to individually analyse each of the store groups and the individual stores within each group. The hierarchy makes sense as the relation parent and child is one to many.

Table 6: Store hierarchies

Product dimension table

This table keeps the information of all products in the superstore; it looks at the business from the product perspective. It can be used to study the different products, brands, and find patterns allowing us to figure out which products sell the least or the most, or when they need to be restocked.

- ✓ Table Name: **product**
- ✓ Number of rows: **1559**
- ✓ Fact or Dimension: **Dimension**

Columns	Justification
product_id	Uniquely identifies a row in the product table.
product_sku	Every product has a unique stock-keeping unit code, this attribute stores that code.
product_name	All products have names, this attribute is a reference to their name.
product_brand	A product belongs to different brands, this attribute stores the brand of a product.
product_curr_price	All products have different prices, this attribute stores the current price of a product. This value is subject to change.
product_cost	A product has an acquisition price state by suppliers, this attribute stores that acquisition price through time.
product_weight	Every product has a weight associated with them; this attribute stores the weight of each product in pounds.
recyclable	Products can be recyclable or not depending on their material, this attribute stores a boolean, 1 if is recyclable 0 otherwise.
low_fat	Food products have fat, this attribute keeps track if a food product has low fat.

Table 7: Metadata of Product Table

Product relations

The relation of product towards transactions is **one to many**.

Table Name	Primary Key	Table	Foreign Key
product	product_id	transactions	product_id

Table 8: Entity relation of Product table

Product hierarchies

- ✓ Table Name: **product**
- ✓ Hierarchy Name: **Brands**

Attributes in Hierarchy	Justification
❖ Product_brand ❖ Product_name	This level of granularity allows the product table to put a magnifying glass into the brands and the individual products each of the brands supplies to the superstore. The hierarchy makes sense as the relation parent and child is one to many.

Table 9: Product hierarchies

Time dimension table

As per project request, the **date** is an essential dimension for the company; they want to be able to measure transactions starting from 1997 and including later years as well as the present. This table works as a calendar reference for previous and future time measurements allowing for the superstore to look at their business from the time perspective.

- ✓ Table Name: **time**
- ✓ Number of rows: **a row for every date since 1997**
- ✓ Fact or Dimension: **Dimension**

Columns	Justification
year	Year is a time measurement and is stored in this attribute.
quarter	Quarters are a time measurement and is stored in this attribute.
Month Name	It's an addon attribute for month, it helps in representing time by the name of the month.
Month	Month is a time measurement and is stored in this attribute.
Day name	It's an addon attribute for day, it helps in representing time by the day of the week.
day	Day is a time measurement and is stored in this attribute.

Table 10: Metadata of Time Table

Time relations

The relation of time towards transactions is **one to many**.

Table Name	Primary Key	Table	Foreign Key
time	Date	transactions	transaction_date

Table 11: Entity relation of Time table

Time **hierarchies**

- ✓ Table Name: **Time**
- ✓ Hierarchy Name: **Time Period**

Attributes in Hierarchy	Justification
❖ Year ❖ Quarter ❖ Month Name ❖ Day	This level of granularity allows the Time table to drill down into years, quarters, months and days allowing to reference different time periods within the same hierarchy. The hierarchy makes sense as the relation parent and child is one to many.

Table 12: Time hierarchies

Transaction Fact table

This table keeps the record of products bought, including customer, store, date and region as well as the quantity and price of the transaction. Transactions is a fact table used to explore and analyse sells based on different dimensions and obtain results that allow the company to make business decisions.

- ✓ Table Name: **Transaction**
- ✓ Number of rows: **269,720**
- ✓ Fact or Dimension: **Fact**

Columns	Justification
transaction_id	Uniquely identifies a row in the transaction table.
transaction_date	A transaction date between the store and customer is recorded in this attribute.
customer_id	It allows us to know to which customer this transaction belongs to.
product_id	It allows us to know which product was purchased during this transaction.
store_id	It allows us to know to which store this transaction belongs to.
region_id	If a store where to change location and keep the same id and name but change its location, we would have a problem when analysing transactions. In order to keep a record of the region where transactions took place, I am including it.
quantity	A customer my order the same product one or many times, this attribute stores the quantity the customer purchased during the transaction.
product_price	This attribute stores the price of the product during the transaction. This attribute is not subject to change.

Table 13: Metadata of Transaction Table

Transaction relations

The relation of Transactions towards Time is **one**

The relation of Transactions towards Customer is **one**

The relation of Transactions towards Product is **one**

The relation of Transactions towards Store is **one**

Table Name	Primary Key	Table	Foreign Key
Transactions	transaction_date	Time	Date
Transactions	customer_id	Customer	customer_id
Transactions	product_id	Product	product_id
Transactions	store_id	Store	store_id

Table 14: Entity relation of Transaction table

Transaction hierarchies

There are currently no hierarchies in the transaction table.

Star Schema

The following **PowerPivot** diagram shows a **star schema** representing our tables and their relations. This schema is easy to understand and provides fast response queries with the minimal number of joints needed allowing for this solution to be simple, quick and efficient.

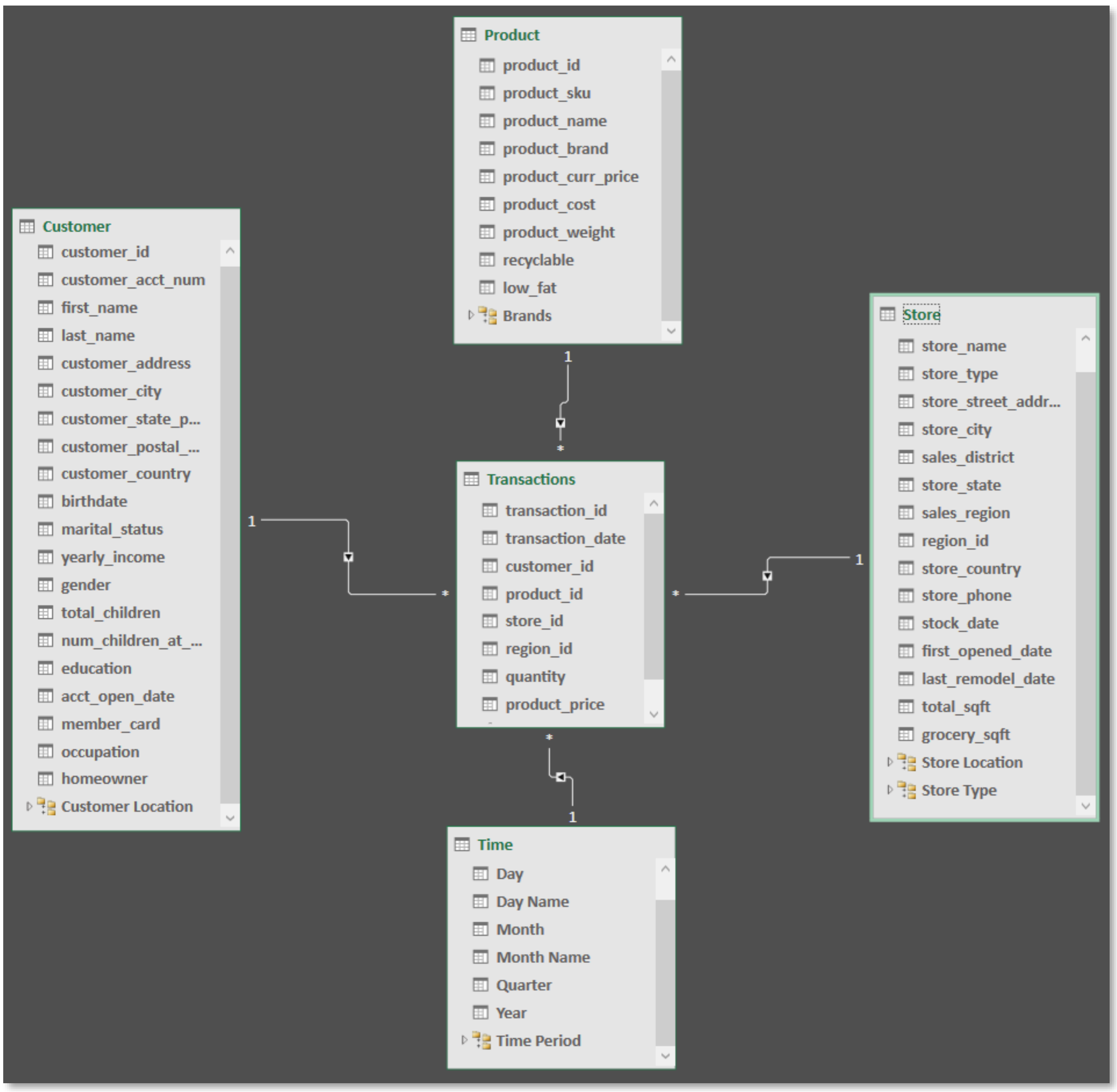


Figure 1: Star schema of my solution. Implemented in Excel power pivot.

As a bonus I have hidden all of the IDs in the transactions table to avoid the end users using the filters backwards.

A Reflection of the Design

One of the discussions in this project was on how the solution would deal with future data. I build a dynamic Time table which expands all its attributes on a new day. This table never stops growing and supporting the other dimensions of the cube when there is new incoming data (new dates).

I have optimized the solution by making a connection with the main table instead of loading it to the environment. A fast-growing database or a big one might cause memory issues; A connection is useful when the tables are updated with new information, this means you can have access to these updates with just a refresh.

You can also connect to a file and have the new upcoming tables information connected and distributed across your design. As an addition, we can create a naming convention so that files names can play a role in the logic behind the tables.

Finally, I have integrated the star schema design allowing me to reduce the complexity of the design, and also improves its analysis thanks to the moderated use of joins within its queries. This means new incoming data will be handled effectively

Task 2

CC & M

The following section will deal with DAX formulas for creating calculated columns and measures in order to analyse my design for handling the superstore data. I have put each formula and the variable name representing its purpose in the following tables.

Table Name	Q #	Variable Name	CC or M	Formula
Transactions	1	Max Retail Price	M	MAX('Product'[product_retail_price])
Customer	2	Average Age	M	AVERAGE('Customer'[age])
Store	3	days_since_opening	CC	DATEDIFF(Store[first_opened_date], TODAY(), DAY)
Store	4	Supermarket_size	CC	SWITCH(TRUE(),Store[total_sqft] > 35000, "Large", Store[total_sqft] > 25000, "Medium", "Small")
Transaction	5	Total Customers	M	COUNTA('Customer'[customer_id])
Store	6	store_street_number	CC	LEFT(Store[store_street_address], SEARCH(" ", Store[store_street_address]))
Transaction	7	age	CC	AVERAGE(Customer[age])
Customer	8	customer_priority	CC	IF(Customer[total_children] >=3 && EXACT(Customer[member_card], "Golden") && EXACT(Customer[homeowner], "Y"), "High-Priority", "Normal")
Transactions	9	weekend	CC	IF(WEEKDAY(Transactions[transaction_date] ,2) >5, "Y", "N")
Transactions	10	Low-Fat Quantity Sold	M	CALCULATE(SUM(Transactions[quantity]), 'Product'[low_fat])
Transactions	11	Total Cost	M	SUMX('Transactions', Transactions[quantity] * RELATED('Product'[product_cost]))
Transactions	12	Total Revenue	M	Total Revenue:=SUMX('Transactions', Transactions[quantity] * RELATED('Product'[product_retail_price]))
Transactions	13	Profit	M	[Total Revenue] - [Total Cost]
Transactions	14	Product Brand Rank	M	RANKX(ALL('Product'[product_brand]), [Profit])

Transactions	15	MTD Profit	M	CALCULATE('Transactions' [Profit], DATESMTD('Time'[Date]))
Transactions	16	Last Month Profit	M	CALCULATE([Profit], ALL(Transactions), DATEADD('Time'[Date], -1, MONTH))
Transactions	17	MoM Profit % Change	M	([MTD Profit] - [Last Month Profit]) / [Last Month Profit]

Table 15: Table with DAX formulas for calculated columns and measures.

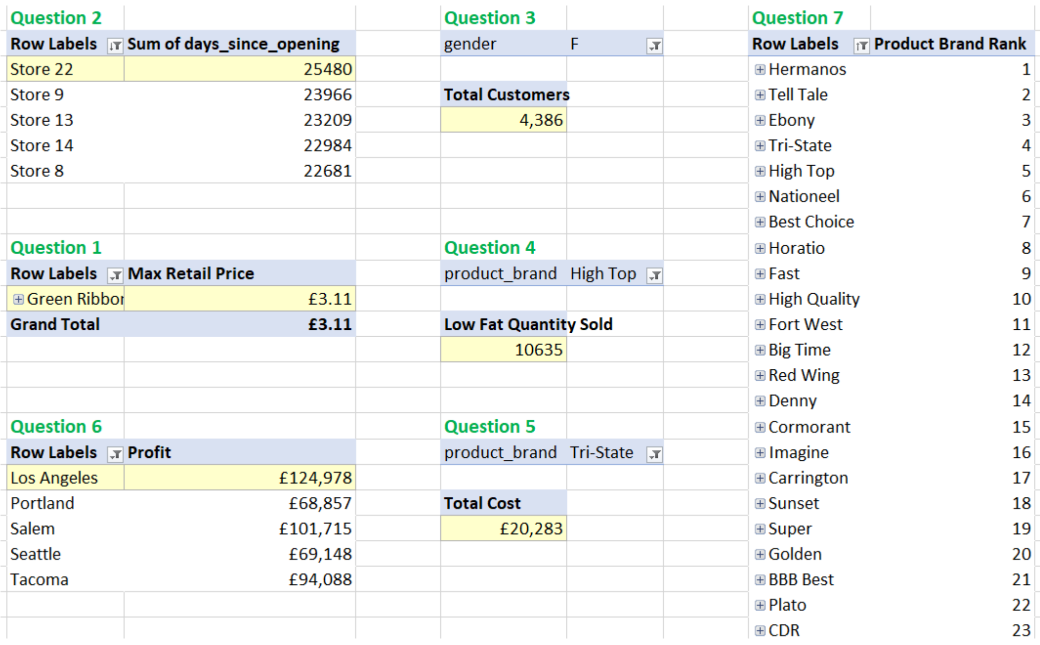
BI Questions

The next task deals with answering business questions which solutions could, in turn, translate into business decisions. The table below contains the questions and the answer I got from my design.

Q#	Question	Answer
1	What is the maximum retail price for the "Green Ribbon" product brand?	£3.11
2	Which store opened first (store number)?	Store 22
3	How many customers are female?	4,386
4	What was the total Low-Fat quantity sold for "High Top" product brand?	10,635
5	What is the total cost of Tri-State products sold?	20,283
6	Which district saw the highest profit?	Los Angeles
7	Which brand is ranked #25?	Bravo

Table 16: List of answered questions

I have shaped these answers into pivot tables just to illustrate and showcase some of my skill and get a different view of our solution. The answer to each question is highlighted in yellow, and I have listed just the top 5 for questions 2,6 and top 25 for question 7.



Question 2

Row Labels	Sum of days_since_opening
Store 22	25480
Store 9	23966
Store 13	23209
Store 14	22984
Store 8	22681

Question 1

Row Labels	Max Retail Price
⊞ Green Ribbor	£3.11
Grand Total	**£3.11**

Question 6

Row Labels	Profit
Los Angeles	£124,978
Portland	£68,857
Salem	£101,715
Seattle	£69,148
Tacoma	£94,088

Question 3

gender	F

Total Customers
4,386

Question 4

product_brand	High Top

Low Fat Quantity Sold
10635

Question 5

product_brand	Tri-State

Total Cost
£20,283

Question 7

Row Labels	Product Brand Rank
⊞ Hermanos	1
⊞ Tell Tale	2
⊞ Ebony	3
⊞ Tri-State	4
⊞ High Top	5
⊞ Nationeel	6
⊞ Best Choice	7
⊞ Horatio	8
⊞ Fast	9
⊞ High Quality	10
⊞ Fort West	11
⊞ Big Time	12
⊞ Red Wing	13
⊞ Denny	14
⊞ Cormorant	15
⊞ Imagine	16
⊞ Carrington	17
⊞ Sunset	18
⊞ Super	19
⊞ Golden	20
⊞ BBB Best	21
⊞ Plato	22
⊞ CDR	23

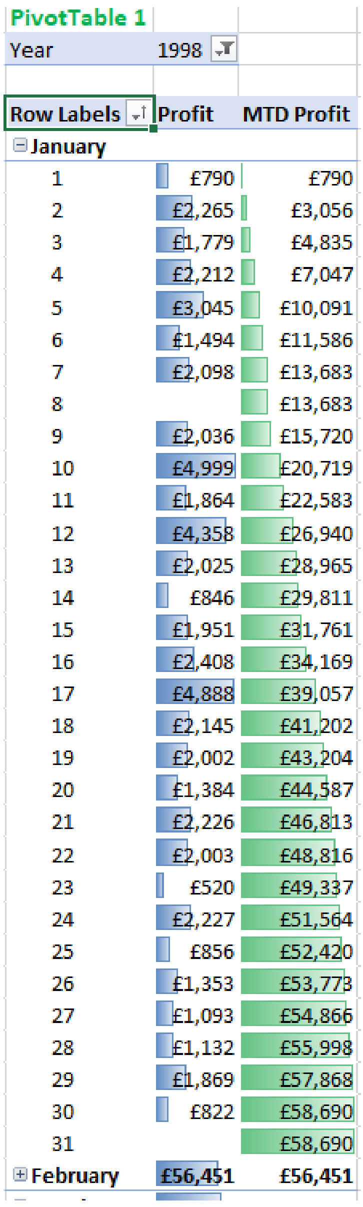
Figure 2: Answered questions as pivot tables

Pivot Tables

PivotTable 1

Should show, for a year 1998 filter, the daily profits and month to date profits. Use data bars to enhance the visualisation of the MTD Profit column.

Figure 3: PivotTable 1

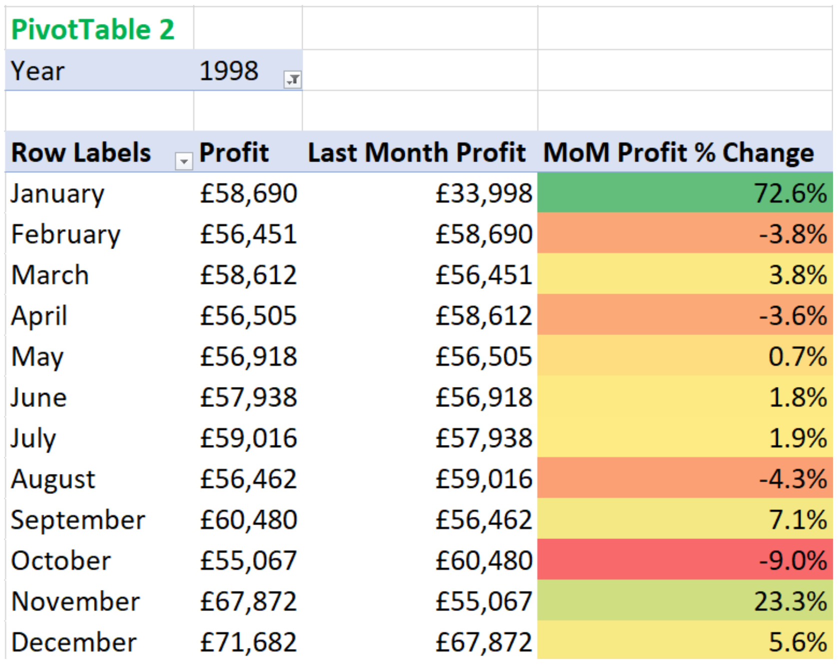


PivotTable 1

Year	1998	
Row Labels	**Profit**	**MTD Profit**
⊟ January		
1	£790	£790
2	£2,265	£3,056
3	£1,779	£4,835
4	£2,212	£7,047
5	£3,045	£10,091
6	£1,494	£11,586
7	£2,098	£13,683
8		£13,683
9	£2,036	£15,720
10	£4,999	£20,719
11	£1,864	£22,583
12	£4,358	£26,940
13	£2,025	£28,965
14	£846	£29,811
15	£1,951	£31,761
16	£2,408	£34,169
17	£4,888	£39,057
18	£2,145	£41,202
19	£2,002	£43,204
20	£1,384	£44,587
21	£2,226	£46,813
22	£2,003	£48,816
23	£520	£49,337
24	£2,227	£51,564
25	£856	£52,420
26	£1,353	£53,773
27	£1,093	£54,866
28	£1,132	£55,998
29	£1,869	£57,868
30	£822	£58,690
31		£58,690
⊞ February	**£56,451**	**£56,451**

PivotTable 2

Should show, for a year 1998 filter, the months of 1998 and their corresponding profits, last (previous) month profit, and percentage profit change of a month compared to the previous month



PivotTable 2

Year	1998		
Row Labels	**Profit**	**Last Month Profit**	**MoM Profit % Change**
January	£58,690	£33,998	72.6%
February	£56,451	£58,690	-3.8%
March	£58,612	£56,451	3.8%
April	£56,505	£58,612	-3.6%
May	£56,918	£56,505	0.7%
June	£57,938	£56,918	1.8%
July	£59,016	£57,938	1.9%
August	£56,462	£59,016	-4.3%
September	£60,480	£56,462	7.1%
October	£55,067	£60,480	-9.0%
November	£67,872	£55,067	23.3%
December	£71,682	£67,872	5.6%

Figure 4: PivotTable 2

Conclusions

My solution for the superstore system has been built based on data warehousing **best practices**. The main table has been decomposed using **OLAP techniques**, this specific technique of **normalisation** has made my design more efficient, quick and has also allowed me to implement the **star schema** which reduced design complexity and optimizes queries. All the **DAX functions** were applied successfully in this structure and techniques for future incoming data have been implemented, allowing for the solution to be also dynamic.

There are a couple of changes that could have been implemented if I had contact with the superstore in order to understand their needs more closely. For example, there are some possible problems with the formulas because they are based on the current retail price, this means is subject to change, there should be another price variable in the transaction table so that it can track the value of all items through time. I was not given enough details to make these assumptions and so I have decided to stay within the scope of this project.

Due to the overall successful results in the implementation of calculated columns and measurements, and the effectiveness of the design, I recommend it as a business intelligence solution.