



An Open Source BI Approach: Concept Proof Tracking Fleet

FÁBIO FERNANDO MORENO LUCENA

Outubro de 2015

An Open Source BI Approach: Concept Proof Tracking Fleet

Fábio Fernando Moreno Lucena

**Dissertation for obtaining the Master's Degree in
Computer Engineering, Specialization in Knowledge-based and Decision
Support Technologies**

Orientador: Paulo Oliveira

Co-orientador: Rui Chambel

Júri:

Presidente:

Vogais:

Doutor Paulo Oliveira Machado

Porto, October 2015

Dedication

To my Family,
Especially to my Maternal Grandmother,
Who fought and Inspired her Grandson.

Acknowledgments

I express my gratitude to my thesis advisor Prof. Paulo Oliveira for his guidance, insight and encouragement throughout the study.

I should also express my appreciation to all GisGeo team members, for the support and opportunity to develop this project.

Moreover, I would like to thanks to my family and friends for the encouragement, patience, sympathy and support during the study.

Resumo

É possível assistir nos dias de hoje, a um processo tecnológico evolutivo acentuado por toda a parte do globo. No caso das empresas, quer as pequenas, médias ou de grandes dimensões, estão cada vez mais dependentes dos sistemas informatizados para realizar os seus processos de negócio, e consequentemente à geração de informação referente aos negócios e onde, muitas das vezes, os dados não têm qualquer relacionamento entre si.

A maioria dos sistemas convencionais informáticos não são projetados para gerir e armazenar informações estratégicas, impossibilitando assim que esta sirva de apoio como recurso estratégico. Portanto, as decisões são tomadas com base na experiência dos administradores, quando poderiam ser baseadas em factos históricos armazenados pelos diversos sistemas.

Genericamente, as organizações possuem muitos dados, mas na maioria dos casos extraem pouca informação, o que é um problema em termos de mercados competitivos. Como as organizações procuram evoluir e superar a concorrência nas tomadas de decisão, surge neste contexto o termo *Business Intelligence(BI)*.

A *GisGeo Information Systems* é uma empresa que desenvolve software baseado em SIG (sistemas de informação geográfica) recorrendo a uma filosofia de ferramentas *open-source*. O seu principal produto baseia-se na localização geográfica dos vários tipos de viaturas, na recolha de dados, e consequentemente a sua análise (quilómetros percorridos, duração de uma viagem entre dois pontos definidos, consumo de combustível, etc.). Neste âmbito surge o tema deste projeto que tem objetivo de dar uma perspetiva diferente aos dados existentes, cruzando os conceitos BI com o sistema implementado na empresa de acordo com a sua filosofia.

Neste projeto são abordados alguns dos conceitos mais importantes adjacentes a BI como, por exemplo, modelo dimensional, data Warehouse, o processo ETL e OLAP, seguindo a metodologia de Ralph Kimball. São também estudadas algumas das principais ferramentas *open-source* existentes no mercado, assim como quais as suas vantagens/desvantagens relativamente entre elas.

Em conclusão, é então apresentada a solução desenvolvida de acordo com os critérios enumerados pela empresa como prova de conceito da aplicabilidade da área *Business Intelligence* ao ramo de Sistemas de informação Geográfica (SIG), recorrendo a uma ferramenta *open-source* que suporte visualização dos dados através de dashboards.

Palavras-chave: Inteligência de negócio, Pentaho, Acompanhamento Frotas, ETL, Armazéns de Dados, Modelo Dimensional

Abstract

Nowadays it is possible to watch a sharp evolutionary process technology throughout the globe. For businesses, whether small, medium or large, are increasingly dependent on information systems to conduct their business processes, and hence the generation of information regarding business and where often the data does not have any relationship therewith.

Most conventional computer systems are not designed to manage and store strategic information, thus making it impossible to support this as a strategic resource. Therefore, decisions are made based on the experience of the managers, when they could be based on historical facts stored by different systems.

Generally, organizations have a lot of data, but in most cases draw little information, which is a problem in terms of competitive markets. As organizations seek to evolve and outperform the competition in decision-making, it arises the term Business Intelligence (BI in this context).

The *GisGeo Information Systems* is IT Company which develops its own software based on GIS (Geographic information systems) using the philosophy of open-source. Its main product is based on the geographical location of various types of vehicles, collecting data, and consequently its analysis (kilometres travelled, duration of a trip between two set points, fuel consumption, etc.). The theme of this project arises in this context, which has aimed to give a different perspective to the existing data, crossing the BI concepts with the system implemented in the company according to its philosophy.

In this project the adjacent BI concepts such as dimensional model, data warehouse, ETL and OLAP process, following the methodology of Ralph Kimball are generally addressed. Some of the main open-source tools are also studied on the market, as well as their advantages/disadvantages in relation to one another.

In conclusion, a solution developed in accordance with the criteria listed by the company, is presented as proof of concept of the applicability of the Business Intelligence at the branch GIS, drawing on an open-source support data visualization tool through dashboards.

Keywords: Business Intelligence, Pentaho, Fleet Tracking, ETL, Data Warehouses, Dimensional Modeling

Index

1	Introduction	1
1.1	Business Overview	1
1.2	Objectives of Project	2
1.3	Structure of Report	3
2	Business Intelligence	5
2.1	Historical Approach of Business Intelligence	6
2.1.1	Fathers of Data Warehousing.....	7
2.1.2	Journey Business Intelligence	9
2.2	Components of Business Intelligence	10
2.2.1	Source Data	11
2.2.2	Extract, Transform and Load (ETL)	11
2.2.3	Data Warehouse	12
2.2.4	Online Analytical Processing (OLAP)	14
2.2.5	Visualizations	15
2.2.6	Dashboards	15
3	Concepts of Data Warehouse	17
3.1	Dimensional Modelling.....	17
3.2	Schema Types	18
3.2.1	Problems in 3 rd Normal Form	20
3.3	MOLAP, ROLAP and HOLAP	21
3.4	Dimension Table Structure	22
3.5	Fact Table Structure	22
3.6	Slowly Changing Dimension (SCD)	23
4	Software Tools	25
4.1	Open Source - An Historical Overview	25
4.2	Main RDBMSs.....	26
4.2.1	SQLite	26
4.2.2	MySQL	26
4.2.3	PostgreSQL	27
4.3	Platforms Business Intelligence	27
4.3.1	Some Available BI Software	28
4.3.2	Open Source BI Software.....	28
5	System Design.....	39
5.1	Related Works.....	39
5.2	Fleet metrics	41

5.3	KPI Analysis	42
5.4	KPI Exploration.....	51
5.4.1	Trip Time & Distance	51
5.4.2	Idle Time Cases	52
5.4.3	Trip STOP Time	54
5.4.4	Fuel Consumption	54
6	System Implementation	57
6.1	ETL Implementation	58
6.1.1	Configuration Connection	58
6.1.2	Extraction Process.....	59
6.1.3	Transformation Process	61
6.1.4	Fact Table	68
6.1.5	Loading Process.....	74
6.2	Main ETL Stream.....	75
6.3	OLAP Implementation	78
6.3.1	Build cube	78
6.3.2	Deploy Mondrian Schema	80
6.4	Pentaho Dashboard CDE	81
6.5	Response to the Requirements.....	85
6.5.1	Software versions	89
7	Conclusion	91
7.1	Limitations.....	92
7.2	Future Recommendations	92

List of Figures

Figure 1 – Gantt Diagram	2
Figure 2 – Understanding the journey of BI [Lachlan, 2014]	9
Figure 3 – Example OLAP cube.....	14
Figure 4 – OLAP Operations Example.....	15
Figure 5 – Star Schema.....	19
Figure 6 – Snowflake Schema	19
Figure 7 – Magic Quadrant of BI tools from Gartern Group[Columbus, 2015]	28
Figure 8 – Agile BI Methodology[Gabelica, 2013].....	30
Figure 9 – Example of ETL process in Spoon	31
Figure 10 – Example of Structure Mondrian Schema Workbench.....	32
Figure 11 – Pentaho Main Menu	33
Figure 12 – Example Saiku Analysis [Barber, 2015]	34
Figure 13 – Top Rated Open Source Based Full-stack BI Software	37
Figure 14 – Idle Time Case 1	52
Figure 15 – Idle Time Case 2	53
Figure 16 – Idle Time Case 3	53
Figure 17 – Idle Time Case 4	54
Figure 18 – Solution Data Flow	57
Figure 19 – Connection Setup: steps to follow (left), wizard window (right)	58
Figure 20 – <i>Table Input</i> Component PDI: visual design (left), menu settings (right)	59
Figure 21 – Text File Input: visual design (left), menu settings (right).....	60
Figure 22 – ETL of Date Dimension	62
Figure 23 – ETL of Time Dimension.....	63
Figure 24 – ETL of Driver Dimension	63
Figure 25 – ETL of Company Dimension.....	64
Figure 26 – ETL of Vehicle Dimension	66
Figure 27 – ETL of Service Dimension	67
Figure 28 – Database Lookup: visual design (left), menu settings (right).....	67
Figure 29 – Extraction and Look up tables (Fact_GPSpart 1).....	68
Figure 30 – APIs Request (Fact_GPSpart 2).....	69
Figure 31 – DimGeography and DimRoad (Fact_GPSpart 3).....	71
Figure 32 – Calculation Fuel Consumption (Fact_GPS part 4)	72
Figure 33 – Loading Fact table (Fact GPS part 5)	73
Figure 34 – Combination Lookup/Update: visual design (left), menu settings (right)	74
Figure 35 – Dimension Lookup/Update: visual design (left), menu settings (right)	75
Figure 36 – ETL Main Job.....	76
Figure 37 – Dimensional Model (simplified)	78
Figure 38 – The structure of Cube.....	79
Figure 39 – Publish Schema Dialog	81
Figure 40 – CDE Layout Structure	82

Figure 41 – CDE Components	83
Figure 42 – CDE Data sources.....	83
Figure 43 – Dashboard	84
Figure 44 – Response to Requirement 1	86
Figure 45 – Response to Requirement 2	86
Figure 46 – Response Requirement 3	87
Figure 47 – Response to Requirement 5	88
Figure 48 – Response to Requirement 6	88

List of Tables

Table 1 – Star Schema vs Snowflake Schema [Diffen, 2013]	20
Table 2 – Slowly Changing Dimension Types	23
Table 3 – Comparison BI Software	35
Table 4 – Proposed Requirement	43
Table 5 – Fields of DimGeography	43
Table 6 – Fields of DimCompany	44
Table 7 – Fields of DimRoad	44
Table 8 – Fields of DimVehicle	45
Table 9 – Fields of DimDate	46
Table 10 – Fields of DimTime	47
Table 11 – Fields of DimDriver	47
Table 12 – Fields of DimService	48
Table 13 – Fields of FactGPS	48
Table 14 – KPI card table explanation.....	51
Table 15 – KPI card of Trip Distance.....	51
Table 16 – KPI card of Time Duration.....	52
Table 17 – KPI card of Idle Time	52
Table 18 – KPI card of Stop Time	54
Table 19 – KPI card Fuel Consumption.....	55
Table 20 – Overview Essential Transformation Process	61
Table 21 – Description ETL process Date Dimension	62
Table 22 – Description ETL process Vehicle Dimension	66
Table 23 – Description ETL process Figure 29.....	68
Table 24 – Description ETL process table Figure 32.....	72
Table 25 – Description of Main ETL process	76
Table 26 – Software Version	89
Table 27 – Experiment Condition.....	89

Acronyms e Symbols

List of Acronyms

GIS	<i>Geographic Informatics Systems</i>
GPS	<i>Global Positioning System</i>
CO²	<i>Carbon Dioxide</i>
POI	<i>Point of Interest</i>
IT	<i>Information Technology</i>
BI	<i>Business Intelligence</i>
GUI	<i>Graphical User Interface</i>
OLAP	<i>Online Analytical Processing</i>
BA	<i>Business Analytics</i>
HTML	<i>Hyper Text Markup Language</i>
CSS	<i>Cascading Sytle Sheets</i>
CSV	<i>Comma Separated Values</i>
URL	<i>Uniform Resource Locator</i>
HTTP	<i>Hypertext Transfer Protocol</i>
SQL	<i>Structured Query Language</i>
UTC	<i>Universal Time Coordinated</i>
RDBMS	<i>Relational Database Management System</i>
API	<i>Application Program Interface</i>
JSON	<i>JavaScript Object Notation</i>
JDNI	<i>Java Naming and Directory Interface</i>
OBD	<i>On-Board Diagnostic</i>

1 Introduction

1.1 Business Overview

The GisGeo Information Systems is a company founded in 2008 which develops solutions with Geographic Information Systems (GIS) integrated in mobile and web, which enable the capture, storage and analysis of geo-referenced data. Its mission is to make their own GIS solutions accessible to business or individuals, focusing on innovation and quality assurance with intuitive design[GisGeo, 2013].

Their solutions are based on open-source technologies with the vision of providing low cost solutions and optimized resources to their clients. Their services are available to individuals or companies all around the world.

The Company is headquartered in Porto and part of its mission is betting on Portuguese talent as well as to build up the economy. During its first year of operation it devoted itself to importation and sale custom localization equipment's.

Only in 2009 did it start to develop its own solutions like fleet tracking using SIG which reference vehicle's geographically and with its own GPS equipment customized installed. Nowadays it has more than 230 clients with its service.

The main product sold is the fleet tracking called GeoCar. This product allows end-users to know the location and vehicle state in real-time in real-time. This system offers real-time communications with the driver, planning the best routes, scheduling maintenance and control CO² emissions. Using an android application (personalized software developed for tablet/smartphones) it is possible for the driver signal POIs (Points of interest) to avoid highways, filling stations and other metrics, which improve performance driving.

1.2 Objectives of Project

Nowadays IT (Information Systems) are very important so that the companies can be successful. Since having structured information is an asset, the business can become more competitive. However to accomplish such goals is not always straightforward given the size of some business and the complexity of implementing systems.

The main goal is to implement a proof concept solution of BI (Business Intelligence), and having the following characteristics:

- Propose a model more efficient for data analysis than the current;
- Improve address data exploration obtained from GPS acquisition;
- Identify a way to calculate fuel consumption of vehicle in vehicles with different GPS equipment installed, even with incomplete data;
- Identify some key parameters that could improve business.

In Figure 1 is possible to have a general vision of the project and how the various tasks were distributed according to the time line available.

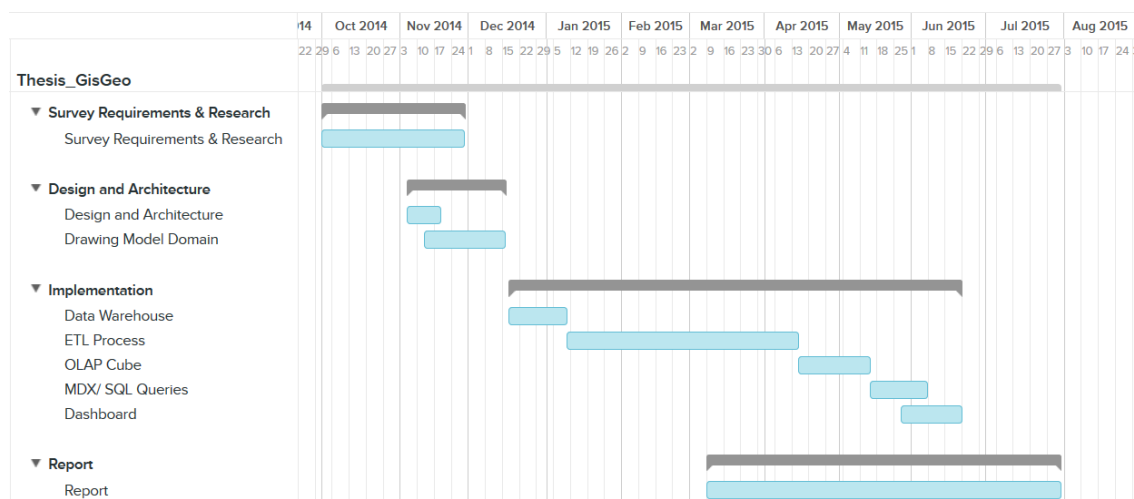


Figure 1 – Gantt Diagram

Fist part of the development was to study the current production database and identify some relevant parameters through literature research. Later on, it was presented a draft data warehouse and refined lately to achieve a possible implementation.

Then, during stage of implementation, ETL process was set to take longer than other tasks since it required a good understanding about data and how could be improved during migration between database from different sources (e.g. complement current gps information with an elevation parameter to allow further analysis). Then after OLAP Cube was generated it was possible to explore combinations between data (e.g. understand which is the road most driven by a determinate company). Meanwhile to give a perspective, it was decided to create a generic dashboard as concept proof.

1.3 Structure of Report

The current report follows the given structure:

- **Chapter 1:** Gives a brief introduction about the theme approached in this project and what structure is followed to achieve the specific goal.
- **Chapter 2:** Generically explains the concepts about BI and consequently the process to build the BI application.
- **Chapter 3:** Gets through some of concepts about Data Warehouse, such as schema types, analysis techniques and the existing mechanism to update the business data.
- **Chapter 4:** It describes the more common open-source databases available in the market as well as the BI software. Furthermore, some of them are explored individually in order to find the most suitable one to be applied in this project.
- **Chapter 5:** Are introduced some relevant aspects that need to be taken into consideration during gathering requirements, since the identification and development of key parameters is part of a successful BI implementation.
- **Chapter 6:** An implementation of the developed solution is proposed, using ETL concepts. Then, is presented the construction of a cube, a dashboard example and the response to the proposed requirements.
- **Chapter 7:** It concludes with an explanation of some difficulties that project has gone thorough during development, as well as the importance of the work done. Moreover, are also presented some limitations and likewise future recommendations that can be applied.

2 Business Intelligence

The definition of the term called Business Intelligence is not an easy one to be delineated. Many of them are centred exclusively on the software used for business intelligence. While the term is often heard in relation to software vendors, there's more to BI than just software tools.

A few examples are shown below identifying possible definitions:

"A variety of software applications used to analyze an organization's raw data." by CIO [Mulcahy, 2007].

"Technologies, applications and practices for the collection, integration, analysis, and presentation of business information." By OLAP.com [OLAP.com, 2015].

"A broad category of computer software solutions that enables a company or organization to gain insight into its critical operations through reporting applications and analysis tools." By Information [Boylan, 2015].

"The use of computing technologies for the identification, discovery and analysis of business data – like sales revenue, products, costs and incomes." By TechoPedia [Janssen, 2015].

Some definitions focus only on software vendors, others on software used for business intelligence or yet fail completely the goal of BI. Therefore, a possible complete definition is:

"Business Intelligence helps derive meaningful insights from raw data. It's an umbrella term that includes the software, infrastructure, policies, and procedures that can lead to smarter, data-driven decision making" [Buys, 2015].

2.1 Historical Approach of Business Intelligence

Since the beginning of mankind the ability to predict future events based on the past was one of the main factors to increase the possibility of survival. So, based on repeated observations of the environment it is possible to discover patterns and draw conclusions. For example, some people, like the Egyptians, Persians, among others, were using this principle. Observe and analyze the behaviour of the tides, the periods of drought and rain, among others, were used to make certain decisions that could help in improving the lives of these people.

When the Industrial revolution and Capitalism came, also new technological advances appeared, leading to increased production and the emergence of new challenges. It was an important moment, since some companies have organized growth and use of information for some resources (people, systems, and machines) to produce more efficiently, thus reaching a product faster and with higher quality. During this enthusiasm of technological developments and ideas, emerged a popular work in 1865 called *"Cyclopaedia of Commercial and Business Anecdotes"* by Richard Millar Devens containing the first known usage of the term *"Business Intelligence"*. He used it to describe the way in which a banker, Sir Henry Furnese, succeeded: he had an understanding of political issues, instabilities, and the market before his competitors. *"Throughout Holland, Flanders, France, and Germany, he maintained a complete and perfect train of business intelligence. The news of the many battles fought was thus received first by him, and the fall of Namur added to his profits, owing to his early receipt of the news."* [Devens, 1864]. As such, the ability to collect and react accordingly based on information retrieved, is today still the heart of BI.

Only in the 20th century did the technology start to advance to the point where it could be considered a landmark. The publication of an article on the subject came out in 1958, written by IBM computer scientist Hans Peter Luhn enhancing the potential of BI, quote Webster Dictionary definition *"the ability to apprehend the interrelationships of presented facts in such a way as to guide action towards a desired goal."* [Luhn, 1958]. At the same time World War II had happened, and the challenge was how to rebuild and improve the various sectors in an organized, simple and fast way. So, new ideas arose and Luhn took part in it by planting a seed with the article titled *"A Business Intelligence System"* [Luhn, 1958] describing an area as "an automatic system...developed to disseminate information to the various section of any industrial, scientific, or government organization" rapidly growing mass of technological and scientific data. With Luhn's work it was possible to expand the possibilities of concept, considering him nowadays as "Father of Business Intelligence".

In the 50s the technology was still uncertain and a distant reality by resorting to the use of punch cards, transistors and COBOL (Common Business Oriented Language). However, most of these technologies were unknown to most businesses. Then, a new technology that would revolutionize our daily lives had begun to take the first steps in the business world. On April 7th, 1953 IBM publicly introduced the first commercial scientific computer [Hope, 2015] and in 1956 the invention of the hard disk revolutionized data storage. Floppy discs, laser discs and other storage technologies allowed to store even more data, than previous old storages types.

So, naturally, in the 60/70s DASD (Direct Access Storage Device) and DBMS (Data Base Management System) arrived, which originated the creation of the first database management systems, collectively referred to as decision support systems (DSS). In the 1980s relational databases appeared; therefore they were much more intuitive for end users; however, complex logic was often required to join multiple tables and obtain the information that was needed. Even though it was possible for end users to write their own simple reports, notwithstanding the queries were often inefficient and had to be run after normal business hours, in order not to impact online transactions. So, at the time a few BI vendors popped up with tools that made accessing and organizing this data possible, yet it was clumsy technology and therefore very difficult to use [Heinze, 2014].

From mid 1980s DSS originated in the computer-aided models, were created to assist with decision making and planning. Through the late 80s the main focus were DSS, data warehouses, Executive Information Systems, OLAP and Business Intelligence.

A 1988 international conference aimed to streamline data processes, held in Rome by an Italian-Dutch-English consortium about Multiway Data Analysis (a method of analyzing large data sets by representing the data as a multidimensional array). The main goal was to reduce the multiple dimensions down to one or two (by detecting the patterns within the data) that could then be presented to human decision-makers, in other words, simplifying BI analysis.

From the point of the 1988 conference, began a modern phase of business intelligence culminating in 1989 Gartner analyst, Howard Dresner, proposing the term BI as an umbrella term to describe *“concepts and methods to improve business decision making by fact-based support systems”* [Power, 2007].

When the 1990's arrived, the companies relied only on the information centres (IC) and data processing centre (DPC) to help executives in decision making. However, as much as they kept the stored data, they offered too little available information. At this point the market had already begun to be more demanding and behaved in a more complex form. This forced to turn their attention to the development of software tools, thus providing more accurate information. The new lead of technology developments, especially data storage, in 1992/1993 raised a large database, which is a major component of BI: Data Warehouse. This new tool improved the flow of data as it moved from operational systems to decision support. One main change was drastically reducing the time it took to access data, besides centralizing all stored data in one location instead of multiple places. Thereby it also lead to the development of other components like ETL tools and OLAP software [Silva, 2012].

2.1.1 Fathers of Data Warehousing

Bill Inmon is considered by many to be the *Father of Data Warehousing*, since he first began the principles about the Data Warehouse and even created the term in the 1970s. Throughout the late 1970s into the 1980s he worked as a data professional improving his expertise in relational Data Modelling. Inmon's work as a Data Warehousing pioneer took off in the early

1990s, when he created his first company (Prism Solutions), where he developed one of the first industry tools for creating and managing Data Warehouse.

In 1992, he published *Building the Data Warehouse* [Inmon, 1992], a book which still is an important part of any data professional's library with a fine-tuned mix of theoretical background and real-world examples. He also developed the concept of the *Corporate Information Factory*, an enterprise level view of an organization's data, of which Data Warehousing, containing servers as repository for Inmon's writing and white papers.

Inmon's approach to Data Warehouse designs focuses on a centralized data repository modelled in the third normal form. It defends using relational modelling of enterprise-wide consistency, which facilitates development of individual data marts to better accomplish the needs of the departments.

Another approach proposed by another father of Data Warehouse was the one from Ralph Kimball. In the book *The Data Warehouse Toolkit* [Kimball, 1996], first published in 1996, he included practical examples for various industries as well as OLAP techniques modeling. His career in IT in 1970s was highlighted by his work as designer for the Xerox Star WorkStation, known as the first company to use a computer with mouse and windows operating system. In 1986 he founded his own company, Red Brick Systems, where he offered products based on relational models suitable for high speed Data Warehousing applications. In 1992 he left Red Brick and decided to start his own consultancy, Ralph Kimball Associates, which is now part of the Kimball Group. In his well-regarded series of books, Kimball addresses topics like web-based Data Warehousing, ETL in a Data Warehousing environment, as well as Microsoft-specific editions that cover SQL Server and the Microsoft Business Intelligence Toolset.

2.1.1.1 Inmon vs Kimball – Different Attitudes concerning Enterprise Architecture

Data Warehousing reached its maturity in the 21st Century as well as schemes between the differing architectural philosophies of Inmon and Kimball. A quote from Inmon emphasizes his mood toward *The Data Warehouse Toolkit* Kimball as "...one of the definitive books of our industry. If you take the time to read only one professional book, make it this book." [Williams, P., 2012].

Inmon's philosophy recommends to start with building a large centralized enterprise-wide data warehouse, followed by several satellite databases to serve the analytical needs of departments (later known as "data marts"). Hence, his approach has received the "Top Down" title. On the other hand, Kimball approach is centred on the development of individual data marts at the departmental level that gets integrated using the Information Bus architecture. Kimball's philosophy is "bottom up", which facilitates star-schema modeling.

Both approaches are today the core of Data Warehousing as standard. However, companies find kimball's data mart approach more suitable with constrained budgets. Dimensional modeling in many cases is easier for the end user to understand, a reason that benefits small companies which do not have an abundance of data professionals on staff.

2.1.2 Journey Business Intelligence

Then, between the late 1990's and early 2000's, business intelligence became a known phrase and was dubbed as *Business Intelligence 1.0*, as showed in Figure 2.

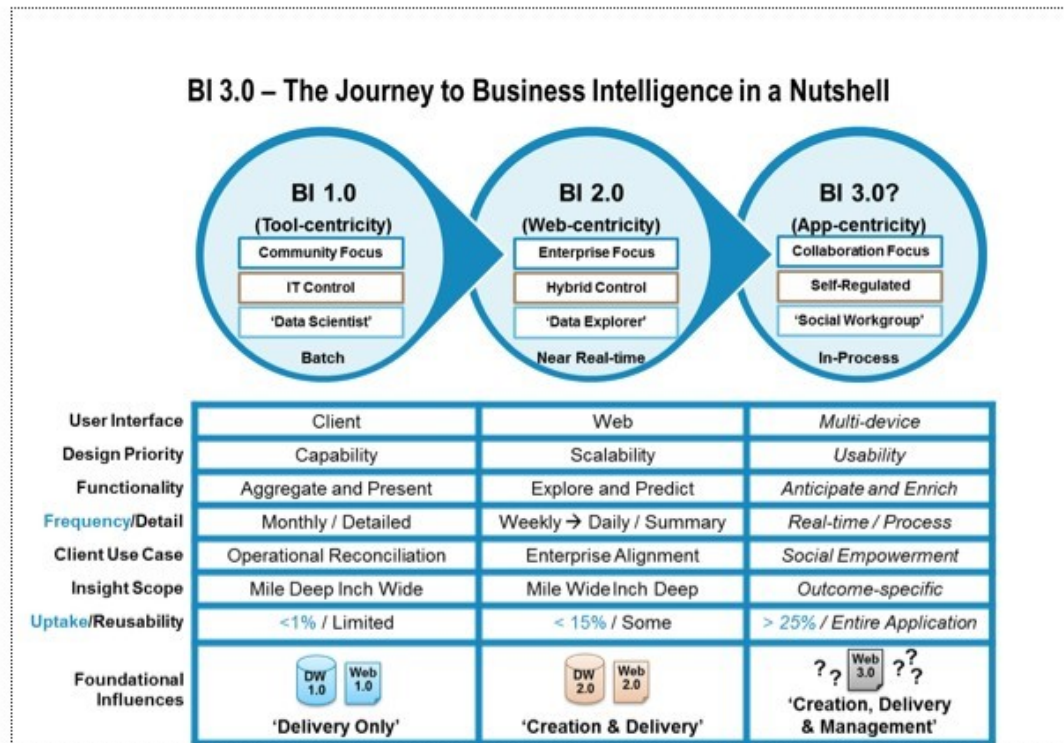


Figure 2 – Understanding the journey of BI [Lachlan, 2014]

During the period of BI 1.0 it was centred on producing data, reports and organizing it and visualizing it in a presentable way. However, two issues still remained: complexity and time. Since most of the projects were owned by IT department most of the users were not capable of executing BI task on their own. Besides, the tools of BI were not fully developed yet, so it was needed to have skilled people with extensive training to gain insights. In addition, data was isolated, which required more time to formulate and deliver reports to the decision makers.

The dawn of the 21st century marked a turning point since the focus was to solve both of the issues: complexity and speed, as well as the beginning of cloud-based programs that expanded and simplified the reach of BI platforms. So, a new era of BI arises known as "*Business Intelligence 2.0*" which hosted different technologies such as real-time processing, which incorporated information from the events as they happened into data warehouses, allowing companies to react more rapidly to events. Another important role was that technological development allowed for non-expert users to develop projects without interference from the IT department [Lachlan, 2014; Heinze, 2014].

The exponential growth of the Internet supported and advanced these developments, due to social networking tools. Blogs, Twitter, Facebook among others, which gave users the ability to express their own ideas and opinions in a simple and fast way. This way of connection between people provided a broad dissemination of BI knowledge. By 2005 the increasing interconnectivity of the business world meant that companies needed real-time information by companies, plus a host other of reasons. Chiefly they needed to keep abreast of the competition, and understand what their consumers wanted and what they thought of their company. So BI was no longer an added utility or a mere advantage. It was becoming a core of successful business in order to stay competitive, and even to remain afloat, in an entirely new, data-driven environment [Heinze, 2014].

The Big Data revolution and explosion of the web left organizations with more data than ever before. It is estimated for example, that over 204 million emails are sent per minute [Heinze, 2014]. It means each person creates an increasingly large amount of information to be analyzed. Since the mid-2000s BI platforms have undergone an intense refining process such as improving tool specifications, expanding self-service options and improving visualization. In the present day, BI tools are designed to fulfil very specific industry requirements, be it healthcare, professional sports, banks, law enforcement, etc. This growth of the industry has contributed significantly to increased adoption of business intelligence. So, one of the main questions stands up: *How to make information easily perceptible?* Visualization tools began to evolve to include the end-user even more. More platforms empowered users to complete self-service access, meaning that they could explore and use their data on their own, without training.

Nowadays more and more companies rely on cloud BI, which hosts the software on the Internet, reducing storage costs and making access to organizational data and insights faster and more conveniently, apart from being in most cases cheaper to maintain. Other mainstream is the rise of mobile-empowered platforms, which allow users to work with BI on-the-go on smartphones, tablets, and other devices. Being simpler and more convenient, these tools encourage wider adoption. At some point it will probably be considered as “*Business intelligence 3.0*” with a predominantly app-centric approach to BI, identifiable by an anywhere, anytime and device or platform independent collaborative methodology. This approach is based on social workgroups and supports self-guided content creation, delivery, analysis and management [Lachlan, 2014].

2.2 Components of Business Intelligence

The various components of BI strategies are Data Warehousing, Data Source, OLAP (Online Analytical Processing), Advanced Analytics, etc. All these data sources can be grouped under three main pillars of BI tools [Surendar, 2014]:

- **Components that aid information and knowledge discovery:** These are categorized by the fact that they help in data extractions from pre-existing data. For example, Ad hoc queries, OLAP, Data Mining and Analytics.
- **Components that analyze data and improve decision making:** These intelligence components are designed to provide automated decision making capability. For example, Business Analytics, DSS, Intelligence Systems.
- **Components for visualizing complex data relationship:** Graphically or visualizing the data analysed what the components under this category manages with. For example, Dashboard, GIS, Visual Analytics.

In order to turn raw data into actionable information, a few of the core components of typical business intelligence development is required. Thus, this section enumerates the basic ones.

2.2.1 Source Data

This is the first step where everything begins, which is the data. Nowadays, the business is in contact with various sources of information that come from transaction systems like CRM systems, ERP systems, inventory databases and payroll systems, and many others. Another example is social media where a company tries to understand the vision of its partners and his influence on the market. Depending on the type of business, it can also be important to include public data from government reports, weather information or industry news reports [Kimball and Ross, 2013].

2.2.2 Extract, Transform and Load (ETL)

ETL is a key part of BI tools and represents the process where data are prepared for analysis. The main problem of data is that it comes from heterogeneous sources so it is likely that not everything is in the same format. To get a full picture of the entire business, it is then necessary to create and apply standards so that the information can later on be analyzed from different perspectives.

Another interesting point is the quality of the data which must be verified before performing any business intelligence. Decision making will ultimately lead to inaccurate results if the data contain errors, it is necessary that the data respect a “single version of the truth” [Buys, 2015].

This process is distributed in three stages called Extract, Transform and Load (ETL):

- **Extract:** information comes from one or more different databases, text files and other sources. The extraction process may include the task of validating and discarding data that does not match expected pattern rules.
- **Transform:** searches to transform data with the purpose of meeting the business and technical needs required on target. Transformation implies tasks such as converting data types, doing some calculations, filtering irrelevant data and summarizing. As an

example, put everything into a standard format (e.g. convert date to the same format, convert gender from 'F' / 'M' and '0' / '1' to true or false, etc.)

- **Load:** After transforming data, the next step is to load it into the target database. Depending on the requirements, the loading may overwrite the existing information, or may add new information each time it is executed.

The transform stage is very important and time-consuming, since it usually takes into account the variety of sources that uses systems from different vendors, diverse types of hardware, and managed by different employees. Moreover, a lot work is required to gather all the data in order that it makes sense together.

2.2.3 Data Warehouse

The data sources are usually transactional systems, so they are designed and built to use data when performing specific functions. Data warehouses, on the other hand, are designed precisely for analysis, allowing the use of all records from all sources at the same time to answer questions. In other words, a Data Warehouse is a repository which contains all combined information from all external sources as well the business applications and systems allowing to analyse them together [Buys, 2015].

The main reason why the data are loaded into this repository happens because they are projected to perform analysis instead of processing transactions. Otherwise, analysing data within separated sources would take too long and could cause disrupt critical business operations.

Secondly, the main point of these systems as stated earlier is to get more insight around the organization in order to obtain a single view of what is going on in the company.

2.2.3.1 Type of Databases

Another reason why these systems are needed is how structured data is stored. The data must be stored differently for analysis compared with the analytical processes, which implies migration from relational to multidimensional database [Kimball and Ross, 2013; Buys, 2015].

A *relational database* is characterized by having a two-dimensional structure. The data are organized using rows and columns in order to be normalized so each attribute can be put in proper place and the entries can be sorted.

On the other hand, a *multidimensional database*, stores data based on more than two dimensions. Unlike existing only by rows and columns like in a spreadsheet, each entity can have various attributes and can exist independently of the other entries.

For example, let's consider a relational database which has a table listing different products in rows and the number of units sold by each state in the column. If the user decides to see other attributes like sales by month, it could require another table and/or create a query.

Alternatively, in a multidimensional model, each product can be explored by its own entity as sales in each state, sales per month, etc. [Kimball and Ross, 2013].

2.2.3.2 Data Marts

Another key concept relevant to BI is the data mart. Essentially, a data mart is considered to be a small part of data warehouse which focuses one particular area on data. A data mart can be broken up according to different operational areas, for example, holds only one subject area as finance, or sales. Data marts that exist as part of a larger data warehouse are called dependent data marts. It's also possible to use standalone data marts, in which the different data marts are not connected [Kimball and Ross, 2013].

However, for companies that want to do enterprise-wide business intelligence, dependent data marts are usually the way to go because the complete data warehouse can hold all the necessary information, while the individual data marts can still be used to address specific needs within the organization. The main reasons for implementing this kind of structure is [Kimball and Ross, 2013]:

- To end-users data marts are less complicated to use for their mostly daily operations;
- Since data marts contains less data, queries of end-users are much quicker;
- Data marts are more specialized, consequently data transformations and integration tasks are considerable faster than DW;
- Building a data mart is a cheaper task in terms of time, resources and feasible solutions than a DW, because data mart is more explicit.

Even though data marts have advantages over DWs, some issues may arise when we are building one, like [Kimball and Ross, 2013]:

Size: In most cases data marts are considerably smaller than data warehouses in size and complexity, but sometimes can match as equal small cooperate DW. A disadvantage is the decrease of queries performance when they start to grow exponentially in the system.

Load Performance: Tasks like loading data and response time become critical when it is verified that a lot of summary tables and aggregations are present.

User access to data in multiples data marts: An often recommended solution is to build virtual data marts which are views of several physical data marts.

Administration: When the amount of data marts increase, it is essential to maintain the data marts activities' coordinated such as versioning, consistency, integrity, security and performance tuning.

2.2.4 Online Analytical Processing (OLAP)

While the ETL process and data warehouse represent the back end of business intelligence, Online Analytical Processing is considered the front end. Specifically, OLAP tools allow users to access, run queries and report on data stored in the system according through various criteria.

For example, if a user wants to see a comparison of products that were bought by Client1 in September versus those that were bought by Client2 at the same time, OLAP can process the information to display it.

The feasibility of many business intelligence applications is to provide capabilities for complex analysis and trend modelling. Whereby the data is stored in multidimensional databases, each attribute of a record is stored as its own dimension in the database. This model offers then much flexibility in making comparisons, tracking trends and looking at data from diverse points of view. These multidimensional databases are often referred as OLAP cubes [Buys, 2015].

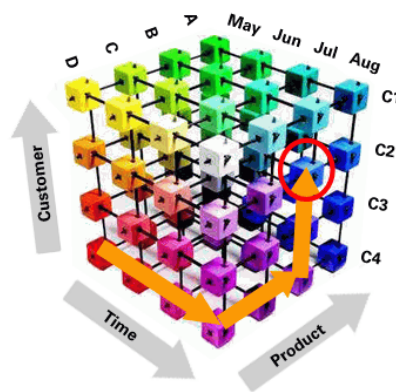


Figure 3 – Example OLAP cube

To perform analyses on the cube, different methods of manipulating data are used, which include [Kimball and Ross, 2013], as exemplified in Figure 4:

- **Slicing:** Select a subset of a cube, creating a new one with fewer dimensions. Usually it is used to isolate some of the criteria necessary for a given query. For example, for product data, if it removes Time Dimension, only the analysis between product and customer remains, considering Figure 3.
- **Dicing:** Specific values are pulled out from multiple dimensions, producing a small cube. For example, the user may only need data about specific product categories.
- **Drill down/Roll up:** Consists in moving through levels of data, going from the most detailed sets to more summarized sets of data. For example, one level can indicate products by each year, but if the user drills down, he can see details for each month. Roll up is the inverse of the drill down process. It applies an aggregation of the data cube in order to reduce the detail information. Basically, it goes up in level/hierarchy.

- **Pivot:** The pivot operation is also known as rotation. It rotates the data axes in view in order to provide an alternative presentation of data.

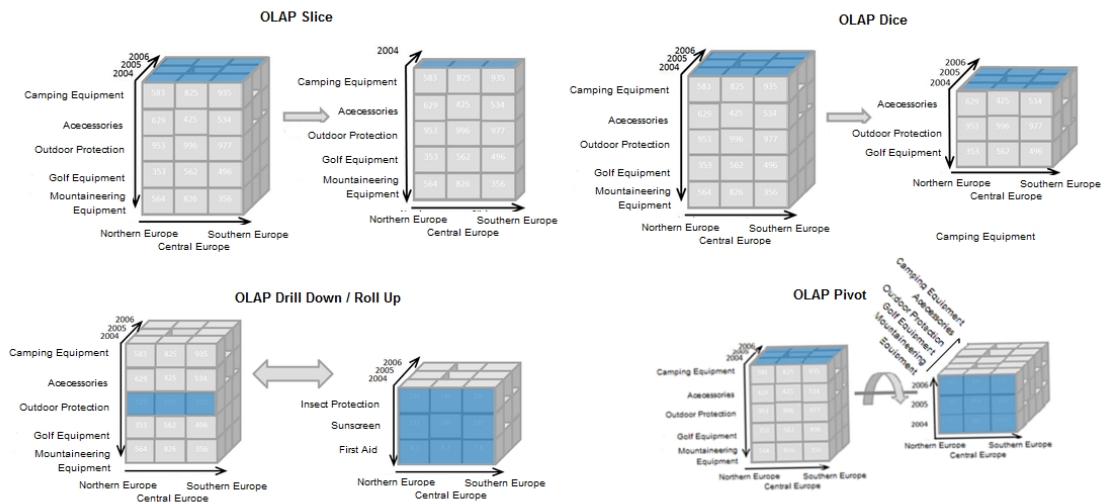


Figure 4 – OLAP Operations Example

2.2.5 Visualizations

One of the goals of BI is to make data accessible and useful to non-technical business users. To them it is not sufficient to show only spreadsheets and a list of numbers, but rather in proper way readily apparent. The tools used for visualization are charts, graphs and other formats. Traditional formats include bar graphs, pie charts and scoreboards, while advanced data visualization can offer interactive and dynamic content that adapts optionally to the user's preferences to represent data [Buys, 2015].

In this category there is also an honour mention to pivot tables, which are worksheet tables that allow for summarizing and analyzing Excel data. When it is needed to look at hundreds of rows of data, it is very difficult to find any common trends or to understand the correlation between your data elements. Using a pivot table allows to effectively analyse your data, since it is possible to obtain different views of original data, since that can be achieved faster than with a report [Providence College, 2014].

2.2.6 Dashboards

The dashboard shows snippets of information the user can review quickly, with the option to choose different items for more detailed reports and visualizations. For example, a head of sales or marketing may log in to see a dashboard that includes a map of where leads are

geographically located, a chart showing the source of leads, graphs showing the average cost per lead for each channel, etc.

A definition of what a Dashboard means is given by Stephen Few's, the leading expert on information dashboards, who defines it as follows: *"A dashboard is a visual display of the most important information needed to achieve one or more objectives; consolidated and arranged on a single screen so the information can be monitored at a glance."* [Taylor, 2014].

By examining the previous state, some key aspects should be kept in mind. Firstly, it should only show what is relevant to the user in order to provide the easiest intelligence before action, in other words, be objective. Secondly, it needs to be customizable, giving the user some control over what information they want to see, either by allowing them to customize the dashboard on their own or by getting their input when dashboards are designed. Thirdly, and last point, it is important for the dashboards' design to look appellative, by presenting information clearly in order to use an effective and accurate visualization [Buys, 2015].

3 Concepts of Data Warehouse

3.1 Dimensional Modelling

All the modelling techniques give different ways to store the data, where each one has its own advantages. For example, ER Modelling stores data in such a way that there is less redundancy. Dimensional modelling, on the other hand, stores data in order to make it easy to retrieve from database. For this reason, dimensional modelling is used mostly in data warehouses built for reporting, but not so good to reduce storage space, reduce redundancy, speed-up loading time, etc. [Mitra, 2015].

According to Ralph Kimball [Kimball and Ross, 2013], dimensional modelling is a design technique for databases intended to support end-users queries in a data warehouse, oriented to understand ability and performance. Therefore, he defends there are four key decisions made during the design of a dimension model:

1. Select the business process
2. Declare the grain
3. Identify the dimensions
4. Identify the facts

The answers to these questions are determined by considering the needs of the business, allowing the design team to determine the tables and column names, sample domain values and business rules.

Business Processes are the operational activities performed by each organization such as processing an insurance claim, registering students for a class, or snapshotting every account each month. It consists essentially in events generated or captured performance metrics that translate into facts in a fact table. Describing each business process allows to define the grain, dimensions and facts that lead to construction of data warehouse bus matrix. The enterprise data warehouse bus matrix consists in rows which identify each business process and columns

with several dimensions. The shaded cells of the matrix indicate whether a dimension is associated with a given business process [Kimball and Ross, 2013].

Grain is the business definition of what a single fact table record represents. It must be declared before choosing dimensions or facts because every candidate dimension or fact must be consistent with the grain. This consistency enforces a uniformity on all dimensional designs that is critical to BI application performance and ease of use. Each proposed fact table grain results in a separate physical table; different grains must not be mixed in the same fact table. For example, a business model of the bank account, has grain as accounting opening value, withdrawal value or deposit value [Kimball and Ross, 2013].

Dimensions is the third step in the design process and the foundation of fact table. Typically dimensions are nouns like date, store, inventory, etc. Moreover, it is where each property or attributes are identified because it will lead to the decision of what columns in each dimension is required [Kimball and Ross, 2013].

Facts or measures are the records which will populate each fact table record. They are the result from a business process event and are almost always numeric, such as quantity or cost per unit. One important task is to understand the granularity of each measure, since it refers to the lowest (or most granular) level of information stored in any table. If a table contains sales data for each and every day, then it has a daily granularity. If a table contains total sales data for each month, then it has monthly granularity [Kimball and Ross, 2013].

3.2 Schema Types

In designing data models for data warehouse / data marts, two schemas types are often used: Star Schema and Snowflake Schema. Figure 5 represents a common data model for relational data warehouses called Star Schema, which consists of one or more fact tables in the middle referencing any number of dimension tables that surround it, forming a star.

In this schema everything has the same level of granularity (the lowest level of information which is stored in the fact table). Besides, is generally more denormalized than model 3FN, thus allowing a better query performance since it is not necessary to do so many joins [Kimball and Ross, 2013].

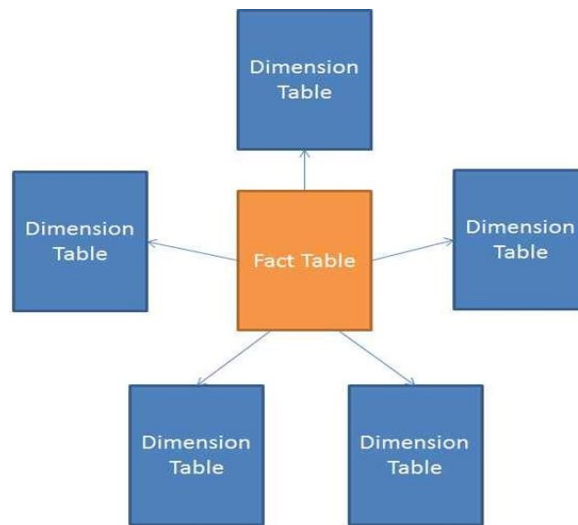


Figure 5 – Star Schema

Another schema is a snowflake, as illustrated in Figure 6, is an extension of the Star schema where each point of the star is shared by other points. The difference between star schema and this schema lies on their level of normalization, typically in 3NF or in higher forms of normalization (e.g. 4NF or 5NF) [Kent, 1982]. So, each dimension table is normalized into multiple tables where each representing a level in the dimensional hierarchy. As a result, the data for a given dimension is spread out over multiple tables, and a diagram of the database looks like a snowflake [Kimball and Ross, 2013].

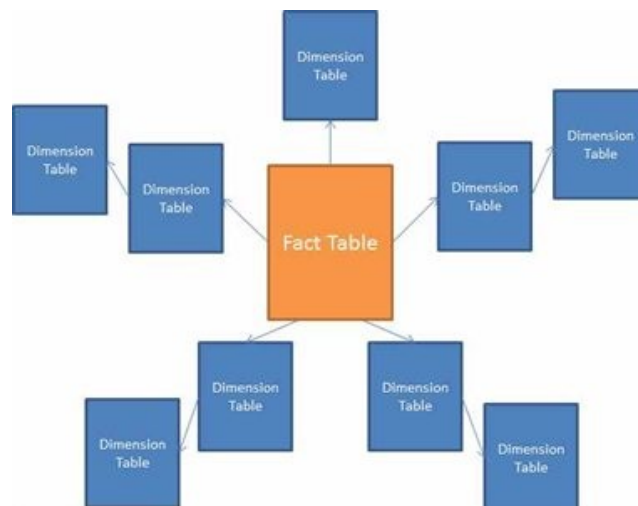


Figure 6 – Snowflake Schema

So, when choosing a database schema for a data warehouse, **snowflake** and **star schemas** tend to be popular choices. However, this comparison discusses suitability of star vs. snowflake schemas in different scenarios and their characteristics. Therefore, a comparison chart is presented in Table 1.

Table 1 – Star Schema vs Snowflake Schema [Diffen, 2013]

	Star Schema	Snowflake Schema
Ease of maintenance / change	Has redundant data and hence less easy to maintain/change	No redundancy, so snowflake schemas are easier to maintain and change.
Ease of Use	Lower query complexity and easy to understand	More complex queries and hence less easy to understand
Query Performance	Less number of foreign keys and hence shorter query execution time (faster)	More foreign keys and hence longer query execution time (slower)
Type of Data warehouse	Good for data marts with simple relationships (1:1 or 1:many)	Good to use for data warehouse core to simplify complex relationships (many:many)
Joins	Fewer Joins	Higher number of Joins
Dimension table	A star schema contains only single dimension table for each dimension.	A snowflake schema may have more than one dimension table for each dimension.
When to use	When dimension table contains less number of rows, we can choose Star schema.	When dimension table is relatively big in size, snowflaking is better as it reduces space.
Normalization/ De-Normalization	Both Dimension and Fact Tables are in De-Normalized form	Dimension Tables are in Normalized form but Fact Table is in De-Normalized form
Data model	Top down approach	Bottom up approach

3.2.1 Problems in 3rd Normal Form

In operation databases the 3rd normal form (3FN) is often used since normally anomalies do not exist with inserting or updating operations; besides, the data has greater independence. The model in 3FN is a technique used to remove data redundancy, assuring data consistency allowing to save disk space. However, this technique involves creating many tables in order to explain each subject, which inevitably leads to join tables decreasing the result performance [1KeyData, 2015].

To understand dimensional data modelling, some terms are explained:

- **Dimension:** Category of information. For example, the time dimension contains details about time.
- **Attribute:** Represents a unique level within a dimension. For example, Day is an attribute in the Date Dimension.
- **Hierarchy:** Is the term which specifies the levels where each represents a relationship between different attributes in the dimension. For example, one hierarchy in the Date Dimension is Year -> Quarter -> Month -> Day.

A dimensional model also includes Fact and Dimension Tables. The Fact tables are usually connected to one or more Dimension Tables, while fact tables do not have a straight forwarded relationship with each other.

3.3 MOLAP, ROLAP and HOLAP

OLAP: An OLAP cube is a logical structure that defines the metadata. The term *cube* describes existing measure groups (which are all combined) and dimensional tables. A *measure group* is a group of measures that matches the business logic of the data and is another logical structure that defines metadata so that client tools can access the data, where each one contains the detail values that are stored in the fact table, which can be copied or dynamically retrieved [1KeyData, 2015; TechNet, 2015].

Multidimensional Expressions (MDX) is a query language for OLAP database, much like SQL is a language for relational databases. For example, a MDX query Code 1 illustrates how to do a query in cube:

```
SELECT
    {[Measures].[Store Sales]} ON COLUMNS,
    { [Date].[2002], [Date].[2003] } ON ROWS
FROM Sales
WHERE ( [Store].[USA].[CA] )
```

Code 1 – MDX example [Microsoft, 2015]

The basic MDX queries contains the “*SELECT*” clause which query the column axis with “*StoreSales*” member of Measures Dimension, and the “*2002*” and “*2003*” members of Date Dimension; the “*FROM*” clause specifies the data source “*Sales*” cube and “*WHERE*” clause define a *slicer axis* that only want “*CA*” (California) member of the Store Dimension [TechNet, 2015].

MOLAP: Multidimensional OLAP cube is the more traditional way of OLAP Analysis where data is stored in a multidimensional cube. The clear advantage is performance, MOLAP cubes are built for fast data retrieve, improved by slicing and dicing operations. Another characteristic is that all complex calculations are pre-generated when the cube is created. Disadvantages of MOLAP include that it is limited in the data types that it can store. Since MOLAP cubes conduct all possible calculations during cube creation, however the type and amount of data stored can become limited. MOLAP also costs additional money to implement since the majority of implementations are considered proprietary and would run an up-front cost for an organization to adopt the technology [1KeyData, 2015; TechNet, 2015].

ROLAP: Relational OLAP cube compared to MOLAP can handle large amount of data that the only limitation is in data size underlying relational database, which means, no limitation on data amount; it can leverage functionalities inherent in the relational database: Often,

relational database already comes with a host of functionalities. ROLAP technologies, since they sit on top of the relational database, can therefore leverage these functionalities. As disadvantage, the performance is slow because each ROLAP report is fundamentally a SQL query in relational database, and the query time can be long if the data size is large; it is also limited by SQL functionalities and for that reason it does not fit all needs [1KeyData, 2015; TechNet, 2015].

HOLAP: HOLAP technologies endeavour to combine the advantages of MOLAP and ROLAP. When detailed information is needed, HOLAP allows “drill through” in the cube. If only a summary of the information is requested, then OLAP leverages cube technology for fast performance [1KeyData, 2015; TechNet, 2015].

3.4 Dimension Table Structure

Each dimension table has a single primary key column, which is embedded as a foreign key in any associated fact table. It is usually a wide, flat denormalized table with many low-cardinality text attributes. These attributes are the primary target of constraints and group specifications from queries and BI applications. A dimension attribute domain value are descriptive labels on report [Kimball and Ross, 2013; KimballGroup, 2015].

3.5 Fact Table Structure

A fact table is the first focus of computations and dynamic aggregations arising from queries. The fundamental design of a fact table is entirely based on a physical activity and is not influenced by the eventual reports that may be produced. Every structure of a fact table has always foreign keys for each of its associated dimensions, some numeric measures, as well as optional degenerate dimension keys and date/time stamps. The facts or metrics stored can be additive, non-additive or semi-additive measures [Kimball and Ross, 2013; KimballGroup, 2015].

To characterize all types of fact tables, they are subdivided into [Kimball and Ross, 2013]:

- **Transactional:** It is the most common type, where for each row a record is stored, which means a new row is created when an event occurs. Typically, a transactional fact table holds data of the most detailed level, necessarily increasing a number of dimensions associated with it.
- **Periodic Snapshots:** is used to show the activity of business occurred during regular time intervals. It can display instant measures or accumulated measures through a time period.
- **Accumulating Snapshots:** These tables store all stages of an event in one line which contains various data dimensions. For example, an order has multiple stages until it is fully processed (request a product; see if it is available; send the product; confirm

reception), each represents a milestone. Essentially, in this example, track each complete stage of an order.

The measure existing types are [Kimball and Ross, 2013]:

- **Additives:** Measures can be added across any dimension;
- **Non Additive:** Measures cannot be added (such as ratios)
- **Semi Additive:** Measures can only be added across some dimensions;

A fact table might contain level facts details or facts that have been aggregated (often called summary tables). Another situation which occurs during the design of fact tables is that these may contain no measures or facts, the so called “factless fact tables” or “junction tables”. They are often used for modelling many-to-many relationships or capture events [Kimball and Ross, 2013].

3.6 Slowly Changing Dimension (SCD)

The "Slowly Changing Dimension" problem is a common one particular to data warehousing. In a nutshell, this applies to cases where the attribute for a record varies over time. So, in Data Warehouse there is a need to track changes in dimension attributes in order to report historical data.

According to Kimball [Kimball and Ross, 2013], there are predominantly 3 types (Type1, Type2 and Type 3) of slowly changing dimension. More recently it was extended as hybrid systems which now can be subdivided into 8 categories [Ross, 2013] described in Table 2.

Table 2 – Slowly Changing Dimension Types

SCD Type	Dimension Table Action	Impact on Fact Analysis
Type 0	No change to attribute value	Facts associated with attribute's original value
Type 1	Overwrite attribute value	Facts associated with attribute's current value
Type 2	Add new dimension row for profile with new attribute value	Facts associated with attribute value in effect when fact occurred
Type 3	Add new column to preserve attribute's current and prior values	Facts associated with both current and prior attribute alternative values
Type 4	Add mini-dimension table containing rapidly changing attributes	Facts associated with rapidly changing attributes in effect when fact occurred
Type 5	Add type 4 mini-dimension, along with overwritten type 1 mini-dimension key in base dimension	Facts associated with rapidly changing attributes in effect when fact occurred, plus current rapidly changing attribute values
Type 6	Add type 1 overwritten attributes to type 2 dimension row, and overwrite all prior dimension rows	Facts associated with attribute value in effect when fact occurred, plus current values
Type 7	Add type 2 dimension row with new attribute value, plus view limited to current rows and/or attribute values	Facts associated with attribute value in effect when fact occurred, plus current values

To give a better perspective how the SCD type 2 works an example is presented. By default it requires to add 3 columns to make the mechanism work with the table: *creation_date*, *end_date* and *active*.

For example, let's consider a situation where a new record arrives to the table and there is no previous information stored. Here the current data is selected to be assigned to the field *creation_date*, in the *end_date* a null value (means it never expire) and *active* with a numeric value 1, indicative that is the current value to be used. After that, the record is inserted into the table.

The next case occurs when a record is already stored in the row, and needs to be updated. In this case, the old record, updates column *end_date* with the current date value and the *active* with the 0 value. Therefore, it indicates that the row is outdated. Then a new row is created with changes made, and it's treated as a new record to be inserted into the table.

4 Software Tools

4.1 Open Source – An Historical Overview

Once the company where this project was developed, chooses to develop its software with open-source tools, the issue is therefore addressed in order to explain its meaning.

The philosophy of open source software advocates in his roots the exchange of knowledge and thoughts, which traditionally are found in the scientific field. The distribution is the base of process evolution, which boosts the development of knowledge.

In the early 80's, Richard M. Stallman [Richard Stallman's personal site, 1983] was the first thinker to register that the 4 types of open software could be:

1. Liberty to execute the software by any means.
2. Liberty to study the operation of the program and adapt it to their needs
3. Liberty to distribute copies
4. Liberty to improve the program and make modifications publically available to the community in order for everyone to benefit from it

To support this project, Stallman created in 1984 the "Free Foundation Software" and launched the GNU Project which protected the previous state free licenses. Today it is known as GPL licence.

In 1998, "Open source Definition" was written by the USA citizen Bruce Perens aiming to describe the technical properties of Open Software and to be used as future reference as text founder of the " Open Source Movement". The main objective of this movement is to offer software free from philosophic or politic aspects which are considered harmful to the marketing. On the other hand, the Open Source movement considers the philosophical/

ethical and political environment as an essential part of the movement and one of its cornerstones [Nunes, 2010].

4.2 Main RDBMSs

A relational database management system (RDBMS) is a database management system (DBMS) that is based on the relational model as invented by E. F. Codd, of IBM's San Jose Research Laboratory. Many popular databases currently in use are based on the relational database model, whereas in this section some of them are explored in order to understand which ones fitted the development of this project.

4.2.1 SQLite

SQLite [SQLite, 2015] is an in-process library that implements a self-contained, zero-configuration, server less, transactional SQL database engine. The source code for SQLite exists in the public domain and is free for both private and commercial purposes. SQLite has bindings to several programming languages such as C, C++, BASIC, C#, Python, Java and Delphi. The COM (ActiveX) wrapper makes SQLite accessible to scripted languages on Windows such as VB Script and JavaScript, thus adding capabilities to HTML applications. It is also available in embedded operating systems such as iOS, Android, Symbian OS, Maemo, Blackberry and WebOS, because of its small size and ease of use [Techopedia, 2015c].

It contains an embedded library which offers an amazing set of tools to handle all sorts of data with much less constrained and ease compared to hosted, process based (server) relational databases. Using an application within SQLite, the integration works with functional and direct calls made to a file holding data (SQLite database). Therefore, avoiding communications through an interface of sorts (i.e. ports, sockets), makes SQLite extremely fast and efficient. This database is good for embedded applications (e.g. applications that need portability and do not require expansion like mobile or single-user local applications) or disk access replacement (easily switching applications SQLite to SQL). On the other hand, it is not recommendable to use in multi-user applications or applications which require high write volumes (only allows one single write operation to take place at any given time) [Tezer, 2014].

4.2.2 MySQL

MySQL [MySQL, 2015] is a full-featured relational database management system (RDBMS) that competes with the likes of Oracle DB and Microsoft's SQL Server. MySQL is sponsored by the Swedish company MySQL AB, which is owned by Oracle Corp. However, the MySQL source code is freely available because it was originally developed as freeware. MySQL is written in C and C++ and is compatible with all major operating systems, made up for this reason one of the most popular databases by its rich features and for being used in large-scale database

servers. To beginner users is relatively easy and there are a lot of third party applications, tools and integrated libraries which help working with RDBMS [Techopedia, 2015a].

As advantages, it stands out the fact that it is easy to work with (third-party tools, and others), it is secure, scalable, and speedy. Some known limitations are reliability issues (the way certain functionalities get handled like references, transactions, auditing, etc.) or stagnated development (might be some complaints regarding development process; some fully databases add value on top of the standard MySQL installations such as MariaDB). Another reason not to use it, is MySQL might not have implemented the full SQL standard (integration with RDBMSs is a difficult process) [Tezer, 2014].

4.2.3 PostgreSQL

PostgreSQL [PostgreSQL, 2015] and Ingres, was an earlier effort, where both were developed by a team at the University of California at Berkeley. PostgreSQL did not originally support structured query language (SQL), QUEL query language was used until 1994, when SQL support was added. In 1996, the first official open-source software version of PostgreSQL was release [Techopedia, 2015b].

PostgreSQL supports almost all relational database features and offers a few unusual features that are normally absent in other RDBMS engines. Commonly supported objects include views, stored procedures, indexes, triggers and object-defined data types, in addition to general RDBMS features such as primary keys, foreign key relationships and atomicity. Besides, it tries to adopt the ANSI/ISO SQL standards among the revisions [Techopedia, 2015b].

Since PostgreSQL or Postgres is an open-source, object-relational database management system (ORDBMS) that is not owned or controlled by one company or individual, it is managed mostly through a coordinated online effort by an active global community of developers, enthusiasts and other volunteers. The downside is when simple read-heavy operations are required, which influences the performance ratios [Tezer, 2014] .

4.3 Platforms Business Intelligence

The BI platforms are set of tools that help the users to consult and analyse the present data in a data warehouse, transforming them into useful information for business. In other words, according to web site Gartner, it allows the enterprise to build BI applications by providing capabilities in three categories: analysis, such as online analytical processing (OLAP); information delivery, such as reports and dashboards; and platform integration, such as BI metadata management and a development environment [Gartner, 2013].

4.3.1 Some Available BI Software

Today there is a huge variety of BI tools and it is not always easy to choose the one that applies best to the project. To understand which choices are available, we resort to magic quadrant of analytics platforms from Gartner Group, the company that does research and consulting in the area of information technologies, about the main BI platforms. As it is shown in Figure 7, they are distributed in four quadrants that are annually classified as leaders, challengers, visionaries and niche market.



Figure 7 – Magic Quadrant of BI tools from Gartner Group [Columbus, 2015]

4.3.2 Open Source BI Software

Nowadays, businesses have access to more data than ever, whereas collecting and analyzing that data and turning it into useful information is a big challenge. The problem resides when the moment comes to choose a BI software to develop this project. Therefore, the primary reasons businesses choose to leverage an open source business intelligence solution over a traditional solution include for example [Team, 2015]:

1. The budget conscious, and a “cost free license” is appealing. (Caution: software licenses are not the only costs associated with open source BI);
2. The organization is in itself an open source company, or you have a policy to support open source initiatives whenever possible;
3. The business may be in itself an open source company or it may have a policy to support open source initiatives whenever possible;

Since the organization gives preference to open source tools, it narrows the choice. But after all, some of the tools can rival the paid BI solutions. After some research about BI open source, it was possible to see a trend in software used to develop projects, highlighting at least three [Oketunji, 2011; Imanuel, 2014; Team, 2015]. Based on the research of these three software, a web site was consulted [BetterBuys, 2015], which has made available several reviews about numerous BI major products free and paid.

4.3.2.1 BIRT

Released in 2004, is a part of an open source Eclipse project sponsored by Actuate with contributions from IBM and Innovent Solutions. BIRT has several components like Report Designer and BIRT Runtime. Other three extra components are Chart Engine, Chart Designer, and Viewer. With this is possible do develop and publish reports as stand-alone solutions. The Design Engine API which can be included in any Java/Java EE application, allows to add reporting features into own applications. The BIRT Report Designer has a rich feature set, performs well and scores high in terms of usability and user intuitive interface. The downside is the lack of a Report Server, but by using the Viewer on a Java application server, it has the potential to provide end users with a web interface a render and view reports. Actual version to present date, is 4.4.2 and is licenced under the Eclipse Public Licence, which runs on Windows, Mac and Linux [Birt, 2015].

4.3.2.2 TIBCO

Acquired by TIBCO in April 2014, the company sells the product JasperSoft. It is considered one of the most popular and widely used open source reporting tool in thousands production environments. JasperReport has available Enterprise and Community edition. This bundle contains several components such as the iReport Design, JasperReport Server, JasperReport Studio and JasperReport Library. The Library contains Java classes and APIs which are the core of JasperReport. iReport Design and JasperReport Studio are report designers which use a pixel-perfect approach in viewing and printing its reports. The ETL, OLAP and Server components are a more valuable features provided by JasperReport in enterprise environments [Jaspersoft, 2015]. JasperReport is based on Java, runs on Windows, Mac and Linux. It also has an excellent documentation supported by Wiki, Q&A forums and user groups. The latest release version is JapserSoft Studio 6.1.0 with licence under AGPL [Jaspersoft, 2015].

4.3.2.3 Pentaho

Pentaho offers a suite of open source BI products developed in Java. Named Pentaho Business Analytics offers ETL capabilities, OLAP Services, reporting and data mining. It provides both community and enterprise editions. An enterprise edition contains extra features through annual subscription, not founded in the community edition. Moreover both versions are often enhanced by extra plugging from the company and the broader community and enthusiasts. Pentaho runs on Java Enterprise Edition and can be used on Windows, Linux, and Mac. The latest release to present date is version 5.3, and is licensed under GPL [Pentaho, 2015].

Pentaho Reporting offers features like visual report editor to web platform to render and view reports to end users. Other extras are the printing version in PDF, HTML, etc., security and role management, and the opportunity to email reports to end users. To run reports and view through web-based user interface, has the Pentaho BI Server, which is J2EE application. It provides a strong community with forum, a bug tracker called Jira and other full documentation [Pentaho, 2015].

As its strengths, it is recognized by requiring easy/medium skills to develop a simple project. Since it has already reached some maturity and reliability, it can be used by software programmers, business intelligence fans, early adopters or college students [Pentaho, 2015]. Pentaho distinguishes itself by being more than just a reporting tool, with a full suite of components (data mining and integration), since it uses a philosophy of *Agile BI* (Figure 8). This perspective tries to give answers to the often common problems placed to actual BI tools: elevate complicity of implementing BI solutions; lack of flexibility in creating solutions before of new business solutions; lack of integrated view between concepts, modulation and visualisation of BI solutions and results [Gabelica, 2013].

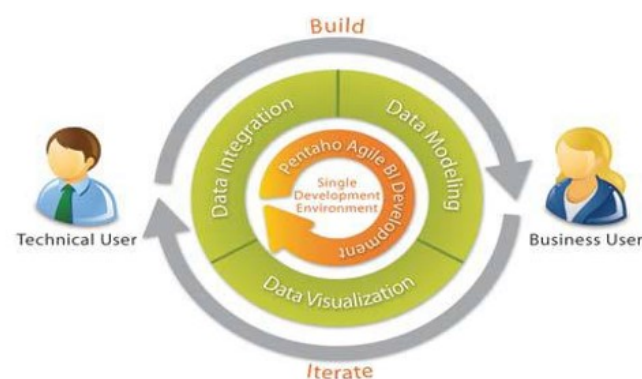


Figure 8 – Agile BI Methodology[Gabelica, 2013]

Based on reviews about user experience [TrustRadius, 2014; G2crowd, 2015] about the previous tools, it was decided to explore a bit more the Pentaho. In next sub-topics are presented summarizes of the most popular products and plug-ins in the Pentaho ecosystem.

4.3.2.3.1 Pentaho Data Integration (PDI)

The software is known as *kettle*, developed by Pentaho; it manages extraction processes, transformation and integration of data in the data warehouse. Kettle is made by *Jobs* and *Transformations*. A *job* can have several transformations and invoke other jobs. It is usually used to migrate data between applications, exporting data from databases to flat files, data cleansing, etc. It supports a vast variety of input and output formats, like text files, data sheets and commercial or free database engines.

Every process is created by a graphic tool (Spoon), thus facilitating the construction of a *transformation* without a writing code (can be called *metadata oriented*). As illustrated in Figure 9, it is possible to see an example of the tool.

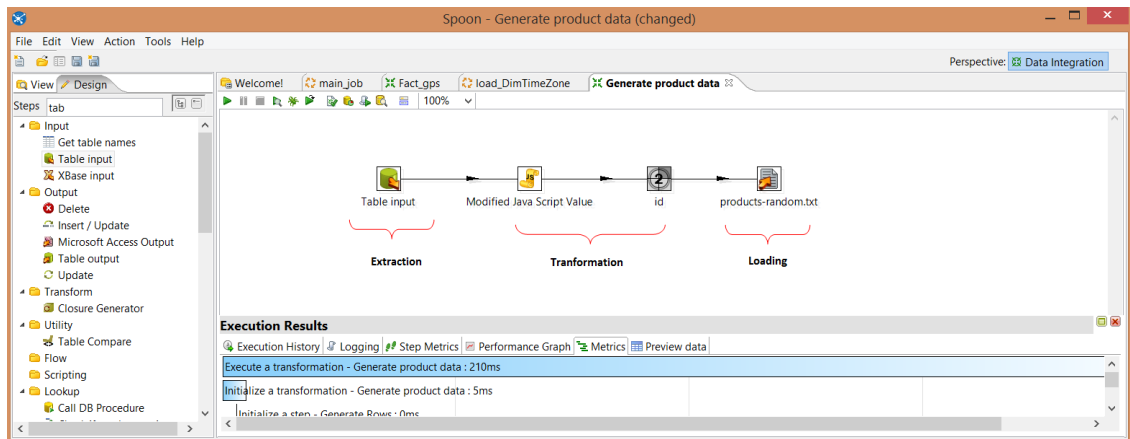


Figure 9 – Example of ETL process in Spoon

Spoon ETL jobs are saved as .kjb files. The “K” stands for “kettle”, which is the ETL engine that runs Spoon jobs, and “JB” is standard for jobs. Thus, for ETL transformations .ktr files, “TR” is an indicative of transformation, and are treated as kettle routines [Holoch, 2014].

The example (Figure 9) shows that data comes from one generic data source and then is stored in a text file. There are no special requirements to build it, is only necessary to understand the function of the component. It is just drag and drop the component and "connected" together using the arrow lines. For more advanced features, it allows to write own transformations using just any language or other packed tool (such R) to accomplish more data manipulation.

On the bottom of Figure 9, Spoon provides a logging monitor about the performance of each component, allowing to understand deviations or delays in the program. On the left side, there is a hierarchical object navigator, which includes database connections, jobs and routines, etc.

4.3.2.3.2 Pentaho Schema Workbench

It is a graphic editor of schemas that allows users to visualise and test OLAP cube (incorporates *Pentaho Analysis Services*, also named as *Mondrian*, which is an open source OLAP server written in Java. It supports MDX query and XML Analysis and olap4j interface specifications. The source files can be in SQL or other format and aggregate data in a memory cache). The schema files are written in XML metadata models that are created with a specific structure by Mondrian Engine. To construct a cube the XML model uses the existing facts and dimensions tables founded in respective RDBMS. The advantage of this tool is that it does not require the actual physical cube to build or maintain; only the metadata model is created, allowing users to edit the model in any text editor. Figure 10 represents the interface GUI [Wood, 2007].

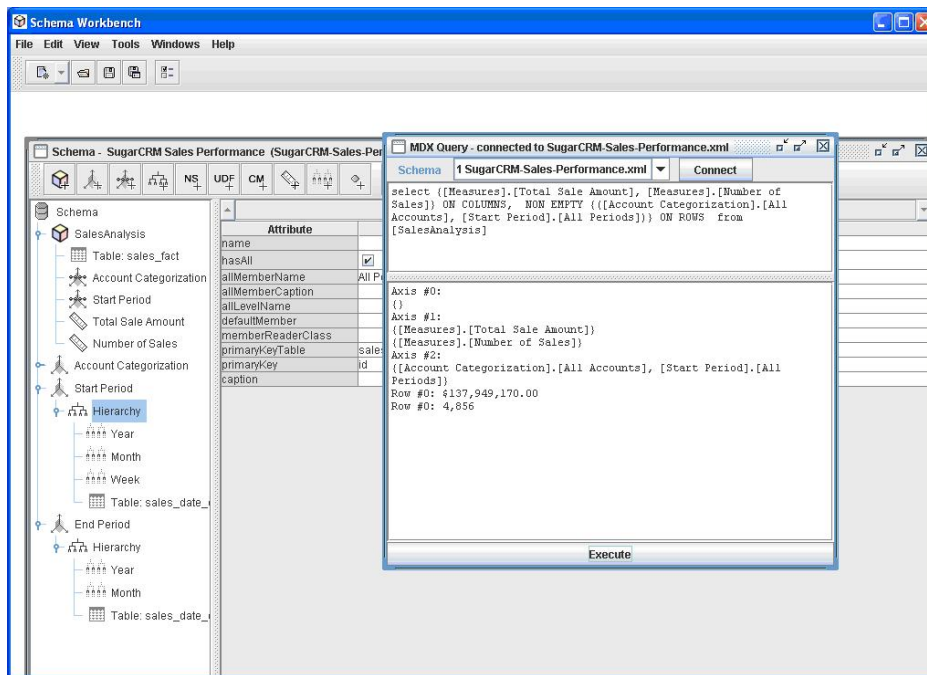


Figure 10 – Example of Structure Mondrian Schema Workbench

In each *Dimension*, there is a collection of hierarchies which discriminate on the same fact table attribute. On the other hand, a *hierarchy* is a set of members organized into a structure (for example, Dimension “Start Period”, an account categorization has year, month, week). A *level* is a collection of members which are at the same distance from the root of the hierarchy (e.g. year, month and week; each represent a level). A *member* is considered a set of attribute values, like year 2014, 2015, etc., or month (January, February, etc.). Finally, a *measure* is associated to the cube and is where mathematical or aggregation operations are applied [Wood, 2007].

4.3.2.3.3 Pentaho BI Platform:

Currently known as Business Analytics Platform, it is the central component to host the content of a BI application. Using a platform it is possible to run and show reports, dashboards, manage security, perform OLAP analysis and many other tasks.



Figure 11 – Pentaho Main Menu

The Pentaho environment (Figure 11) provides 5 perspectives [Ramazzina, 2013]:

- **Home:** Is the basic perspective after login is made with credentials. As it can be seen in Figure 11, it shows *Recent files* and *Favourite Files* allowing fast access to reports or visualizations often used. A blue button *Manage Data sources* is where users can either edit and delete the existing data sources
- **Browse files:** Gives the ability to access different solutions, depending on granted privileges. For example, only a user with Admin role has access to all of the users' home directories. The public part, identified by root folder, is called *Public*. That part is the part of the solution that is shared, depending on the share level decided by the administrator or by the content owner, by every user in the system.
- **Marketplace:** Provides extra plugins for free or paid which allows extend functions of Pentaho (create more professional and visually attractive reports, dashboards, do analysis, etc.). For example, Saiku allows to create reports or dashboards with professional aspect and visually attractive to the end user.
- **Schedule:** This perspective is where the user can check the status of any of the scheduled content and open for any terminated execution.
- **Administration:** Is the location where users / roles are managed according to tasks they execute. Essentially, is where the administrative tasks are done.

4.3.2.3.4 Saiku

Saiku is a modular analysis suite offering lightweight OLAP which can be easily embeddable, extendable and configurable. It was originally founded in 2008 by Tom Barber and Paul Stoellberger with the name *Pentaho Analysis Tool*. After some development reaches its maturity in 2010, reborn as Saiku [Barber, 2015].

Saiku is a web based analytical solution, friendly to user's make quick and fast data analysis by letting creating and sharing reports. The solution allows to connect a wide range of OLAP Servers such as Mondrian, Microsoft Analysis Services, and Oracle Hyperion and can be

deployed rapidly in order to users explore data almost in real time. It offers an interface written in HTML, Javascript and CSS easily modified and adapted to user's preferences (Figure 12). By using RESTful standards, the server can be easily integrated into different user interface technologies and 3rd party applications [Barber, 2015].

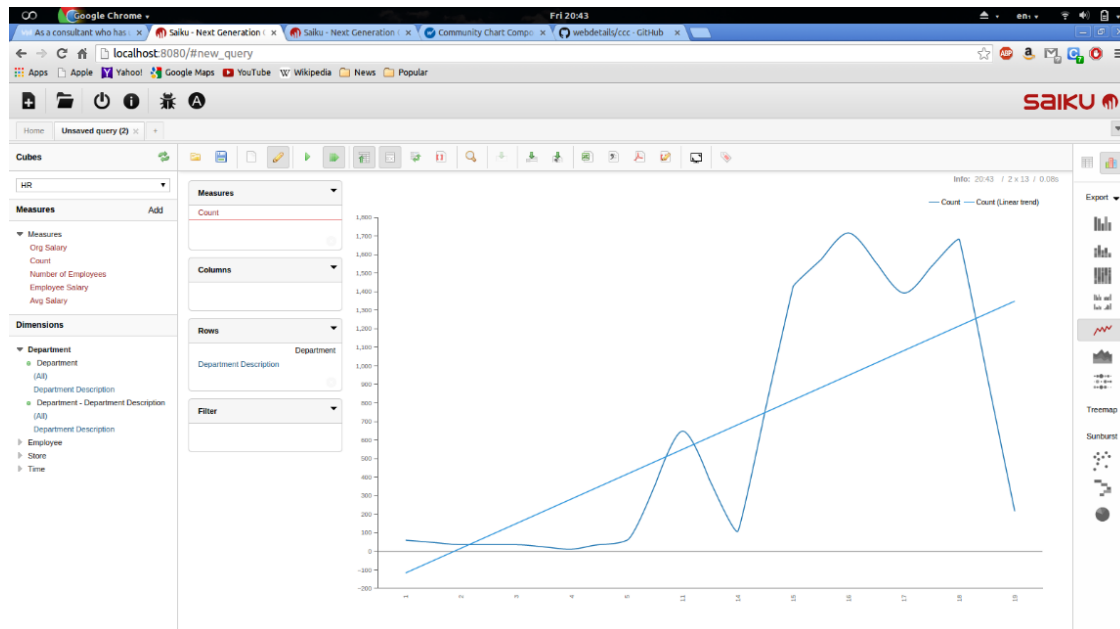


Figure 12 – Example Saiku Analysis [Barber, 2015]

4.3.2.3.5 CTools

Developed to support community edition Pentaho BI server, ctools is a framework that aims to create and maintain dashboards. It provides a set of tools to design integrative dashboards which are integrated in the BI server. The CTools, or tool-sets are [Webdetails, 2015]:

- Community Dashboards Framework (CDF)
- Community Chart Components (CCC)
- Community Data Access (CDA)
- Community Dashboard Editor (CDE)

The CDF is the engine that generates the interactive dashboards and makes them available in platform through the user console.

CCC is a library of charts which is built on top of Protovis, a powerful free and open-source visualization toolkit. Inheriting his properties, allows to users an extensive customization of properties, easing the development process and making it highly interactive.




CDA is a component developed to gather, merge and deliver data from different sources in a uniform manner, even in different languages (e.g. joining data from SQL queries with data from MDX query).

CDE is an advanced editor of dashboards which enables, edition and rendering process at same time creation, where the user can jump backwards and forwards. During the design of dashboard, CDE offers three perspectives [Webdetails, 2015].

- **Layout:** Allows designing the layout of the Dashboard, either from scratch or using some template. Plus, it supports styles and HTTP elements (text or images).
- **Components:** Is a menu which displays components in dashboard, includes text boxes, tables, charts (pies, bars, timelines), radio buttons, OLAP views, etc. Another advantage is the use of JavaScript code that allows customizations about visual and behaviour of components.
- **Datasources:** In Datasource perspective, is the place where data is set up to populate the dashboard. For example, data may come from Mondrian cubes, XML files, databases, kettle transformations (data gather from Web services), etc.

In order to show the comparison between BI tools, the extracted information [Heinze, 2015], as well as a review from users experience [TrustRadius, 2014] is presented in Table 3.

Table 3 – Comparison BI Software

			
Vendor:	Pentaho	TIBCO	Actuate
Product:	Pentaho BI Platform	Jaspersoft	BIRT
Fouded:	2004	1997	1993
Headquarters:	Orlando, FL	Palo Alto, CA	San Mateo, CA
Ownership:	Public	Private	Private
Customers:	1,200+	500+	8 000
Deployment Model:	On-premise, cloud	On-premise, Cloud	On-premise, Cloud
Intended Users:	All	All	All
Free Trial:	Yes	Yes	Yes
What's Unique:	Pentaho offers BI software that distinguishes itself by its ability to unite data integration processes with business analytics. This seamless approach helps users efficiently integrate and visualize key information.	TIBCO Jaspersoft is a business intelligence solution that offers self-service access to an embeddable reporting and analytics platform, helping to speed up the time to insights, and make decisions faster.	Actuate BIRT is a business intelligence tool that is based upon open source BIRT and the Eclipse platform. Actuate is maintained by a very active community of users and the convenience of its features, such as a one-button install and support for production deployments.
Product Details:	Pentaho includes a spectrum of advanced analytics, from	Reporting and analytics can be accessed within an on-premise or cloud app.	BIRT is comprised of two main parts: a report designer

	<p>predictive modeling to basic reporting. The tool is specifically designed to blend with mobile platforms, providing a seamless experience from smartphones and tablets.</p>	<p>Dashboards, visualizations, rich analytics including a web-scale platform, and self-service reports are just a few of the capabilities supported by Jaspersoft, which can be easily embedded in both internal and commercial applications.</p>	<p>based on Eclipse, and a runtime feature that melds BIRT reports with any application. Actuate's BIRT software aims to change that by designing tools that are meant to be used by all, from technologically savvy developers to average end-users.</p>
Features:	<p>Pentaho also has many unique abilities such as powerful visualizations, geo-mapping, heat grids, and scatter charts. The system relies on in-memory data caching, which provides analysis of data at the "speed of thought," making for a vastly quicker BI tool.</p>	<p>One of Jaspersoft's most notable features is its reporting ability, which helps end-users stay swiftly informed and make better business decisions. Their reporting tools draw data from multiple places and display it in a simple, straightforward, interactive way for users to analyze and draw insights from.</p>	<p>Actuate offers three kinds of reporting tools, which each allow developers to integrate data sources and design their own reports with customizable insights. These include BIRT Designer, Designer Pro, and Studio.</p>
Target Market:	<p>Pentaho works with clients like edo, Exact Target, the 9/11 Memorial & Museum, and Lufthansa.</p>	<p>Jaspersoft has a strong presence in the non-profit and medical fields. Some customers include the Sierra Club, the USDA, and the Naval Safety Center.</p>	<p>Actuate works with several different types of industries. Clients include the City of Chicago, City of Dallas, and the British Royal Air Force.</p>
Implementation:	<p>Pentaho offers consulting services which create pre-packaged and custom engagements to help develop and implement tactics quickly and smoothly. An extensive training suite is available, which includes onsite and online offerings.</p>	<p>Users of Jaspersoft can login to an online account on their website where they can create and track requests and investigate support resources to help them in their use of the product.</p>	<p>Actuate's Professional Services team of BIRT consultants offers a number of engagements that facilitate rapid deployment and best practices knowledge transfer on key architectural concepts to ensure end-user adoption and future growth.</p>
Customer Support:	<p>Pentaho's user community has been touted as being one the most informative and helpful on the market, and this is one of the</p>	<p>Paid support options include Self-Service Express Support, Professional Standard Support, and Professional Premium Support. They feature "Self-</p>	<p>Actuate offers customer support on three main levels: by offering support plans online that allow users to</p>

	vendor's best support tools. More hard technical support packages are available, and these include issue detection, problem resolution, and developer assistance.	Service Support" which is a compilation of documents, resource guides, and intuitive search technologies.	navigate through issues on their own, an Open Source Project Support package that must be subscribed to, and a number of global support centers.
About Vendor:	Pentaho Corporation is a business intelligence software company that specializes in open source BI products. It's most notable suite is Pentaho Business Analytics, an offering which includes OLAP services, reporting, data mining, dashboarding, and data integration capabilities.	TIBCO is a software company that offers technology for both on-premise and cloud servers. Founded in Palo Alto, California in 1997, TIBCO has grown from technology incubation tools to a wide variety of business-to-business services which primarily focus on business intelligence.	Headquartered in San Mateo, California, Actuate's most notable products include BIRT Analytics for predictive analytics exercises, BIRT Designers, BIRT iHub for developers, and Customer Communications Suite.

Figure 13 covers a comparison between three top open source business intelligence and reporting tools [TrustRadius, 2014].

	TOP RATED		
	PENTAHO	JASPERSOFT	ACTUATE
Likelihood to recommend	8.0	7.8	7.6
Likelihood to renew	8.9	8.3	6.9
Product usability	5.0	NA	NA
Product availability	NA	NA	NA
Product performance	NA	NA	NA
Support rating	8.0	7.0	10.0
In-person training	NA	9.0	NA
Online training	NA	NA	NA
Implementation	5.0	9.0	NA
Data sharing/collaboration	6.7	NA	4.3
Data sources	9.3	NA	7.3
Data visualization	6.3	NA	5.2

Figure 13 – Top Rated Open Source Based Full-stack BI Software

All three of these open source business intelligence and reporting tools provide a rich feature set ready for enterprise use. It will be up to the end user to do a thorough comparison and select either of these tools. Major differences can be found in report presentations, with a

focus on web or print, or in the availability of a report server. Pentaho distinguishes itself by being more than just a reporting tool, with a full suite of components.

5 System Design

5.1 Related Works

The research done to find a suitable model to store the positioning data from moving objects which contemplating fuel consumption, was a challenge. Many have performed research in the field of mining position data, and some of those researches are in one way or another related to that field, but very few have focused on data warehouses with trajectory observation and with environment perspective in mind. Next, are present some of the most important works found, which contributed to the development of this thesis.

The work about *Mobile Information Collectors Trajectory Data Warehouse Design* [Oueslati and Akaichi, 2010], focuses on creating a trajectory data concept model and propose a conceptual model for the Trajectory Data warehouse destined to gather trajectory data into a repository for query analysis triggered by On Line Analytical Processing (OLAP) users. The data results are gathered from mobile information collectors' trajectory, where is analyzed, according to trajectory characteristics, for decision-making purpose. In this paper, the authors introduce several researches related to movement scenarios of moving objects, trajectory data conceptual model and the data warehouse modeling (Data Warehouse, Spatial Data Warehouse, Spatio-Temporal Data Warehouse and Trajectory Data Warehouse). The present work develops a model based on a snowflake schema containing tables to store points of interest and his characteristics (artificial like Educational-company which has a name, type of company, where is located, etc.; Transportation-company, etc. and natural as Mountains containing such as name, length and location; Sea, Lake, etc.) joined with fact table trajectory, with measures about time trajectory. At the end, they defend their work is starting point to make strategic decisions about implanting new commercial activities and finding new opportunities.

Another work *Travel-Time Estimation in Road Networks Using GPS Data* [Jensen, A.F. and Larsen, 2010], states the traffic jam is a huge problem on overpopulated areas. Being able to

predict when congestion occurs ahead of time would enable drivers to choose a different path than originally planned, and it could potentially lead to reduce the amount of hours lost in traffic each day. Collecting traffic data is too expensive and time-consuming, as it requires the use of loop detectors, license plate recognition system, or other kind of software. The solution presented is that the consumer uses GPS products, even though they have less accuracy. So, it addresses the development of a method that the consumer uses GPS products to estimate travel times accurately in a road network. It is achieved by collection groups of GPS points per road segment, which calculates a travel time based on average speed. Thus, presents a data warehouse schema, where GPS observations are stored in fact table supported by dimensions as vehicle types, drivers and road conditions. Plus, it is discusses a strategy how to deal with GPS independent points in order to match them correctly on a map, and the best way to store given the acquisition rate. It is confronting two approaches: work with point-based approach where points are stored independently or trip-based approach which aggregates several points into segments. Based on his studies, they conclude the last one brings more advantages since is more space efficient, more accurate and faster than the point-based approach. At the end, the authors intend with their work, provide a tool for traffic planners and analysts with the purpose of potentiate the flow of traffic in large cities.

In a paper called *An Advanced Data Warehouse for Integrating Large Sets of GPS Data* [Andersen et al., 2014], was one of the cornerstones of this project, since it provided guidelines and proven methods of a successful implementation. It addresses the integration of GPS data with fuel consumption data and weather conditions, applied to Denmark country. Further, implements a purely relation data warehouse in a PostgreSQL DBMS, that handles GPS data from multiple sources (approximate 3.4 billion rows from 16 different data sources). It was used a star schema to relate GPS observations with several dimensions, alike spatial, temporal and spatio-temporal. The data were classified as static, like vehicle information data and dynamic, such as weather data. They acquired the common fields from the vehicle position such as latitude, longitude and timestamp, and the specific ones, like speed, course, quality of signal and altitude. In addition to static data, it also covers three spatial sources integrated into the system (zip codes, municipalities and time zones), load from the Danish Geodata Agency. On the other hand, the dynamic data are considered the weather measurements, a spatio-temporal data source, which is available from a global network of weather stations. This data is dynamic as new data is repeatedly being fetched from external web sources. According to the paper, they improved analysis on speed, fuel consumption, driving pattern, etc. associated with weather conditions (e.g. sun or rainy day, ice roads, wind direction, etc.). Furthermore, shows a variety of optimization techniques applied to DBMS in order to improve performance, such use indexes, column alignment (technique for ordering the columns to minimize the disk usage) or Unlogged tables (data written into unlogged tables is not written to the write-ahead log, which makes them considerably faster than ordinary tables; if the data is non-critical and losing it is acceptable). The trajectory data is stored in HERMES [Pelekis and Frentzos, 2008], a database that supports several spatio-temporal queries, and that computes the trajectory metadata and then loads them into the data warehouse. In conclusion, the development work was partly

supported by Reduction¹, Danish Environment Agency, where the system has been operational since March 2011 and is used for both research and commercial purposes.

5.2 Fleet metrics

Whether they know it or not, fleet managers use metrics every day. They measure performance, provide benchmarks, and offer goals and targets at which the day-to-day and strategic activities of the fleet function can aim. Just about any activity can be measured using metrics, like batting average in baseball is a metric, miles per hour is a metric measuring speed, units of weight, size and distance are metrics. Some are more useful than others.

So, establishing key performance indicators (KPI) or metrics of vehicle assets, and vendors, permit effective fleet performance. This allows chief decision makers to observe trends as they evolve over time, affecting organization's overall profitability. The following indicators are some of the key parameters which are commonly analysed [Hatfield, 2014]:

1. **Km Traveled:** The number of km travelled is one area where fleet managers tend to have an interest. In some cases it happens that some trips are not acknowledged to day-to-day business purpose data. Such events will increase vehicle driving costs. A practical solution involves into monitoring driver territories, business-use reports and number of sales/service calls in relationship to mileage and time. With GPS it is possible to reduce unnecessary mileage by providing improved routing, but also discourage excess usage that can occur when the driver believes "no one is looking".
2. **Vehicle size and weight:** With an advance on high strength steel, aluminium, composites and magnesium these lightweight materials play a bigger role when choosing the purchase of the vehicle instead of traditional iron and steel materials. It is necessary to understand business requirements and choose appropriately vehicle models, which avoid the negative aspects such mechanical failure and downtime.
3. **Modifying driver behavior:** This type of analysis is growing since it has considerable impact on fuel consumption. Inconsistent speeds, idling, hard acceleration, excess use of air conditioning and hard ranking are some habits that can be trained and/or corrected. EPA (U.S. Environmental Protection Agency) estimates an increase of fuel efficiency as much as 33 percent [Hatfield, 2014]. However this requires collaborations at all levels of management in order to be a successful goal. Some vehicles have within this report "events" which facilitates this type of analysis.
4. **Reduce Lifecycle Costs:** It is common to see some executives make vehicle replacements as an unnecessary cost to the overall fleet budget, instead encouraging fleet managers to retain their vehicles until they reach an older asset age. Another

¹ Available at: <http://www.reduction-project.eu/>

mistake is to retain and operate vehicles far past their optimum economic life which may lead to excessive maintenance costs, since increase fuel costs as the vehicles decrease in fuel economy, and reduced utilization. A lack of capital funding or failure to communicate the costs and benefits of timely fleet replacement can be an issue. So a good plan is to consider all relevant factors (e.g., initial new vehicle cost, reasonable projected resale value, fuel L/Km, planned maintenance and projected repair, personal use payments), in order to prepare short and long-term replacement plans.

5. **Lower Maintenance Costs:** Fleet managers believe that preventive maintenance (PM) should occur near 5000 km. Such frequent PMs are only required for vehicles that operate under “severe” duty as defined by the Original Equipment Manufacturing (OEM). Therefore, executive decision makers should consult with their fleet managers to determine the ideal practices for the company’s fleet, using OEM recommendations as a resource. Plus, using synthetic oils or the use increased diameters of OEM-specified tires, will allow extending intervals between PMs Maintenance.
6. **Lower Overhead Costs:** Overhead costs, also known as indirect costs, include the cost of management and administrative staff, buildings and facilities, including fuel sites, computer systems, utilities, tools, taxes, and many other factors that cannot be attributed directly to a vehicle. Since there is no actual formula for calculating the percentage of a fleet budget devoted to overhead, Activity Based Costing (ABC) exercise is useful for identifying the sources of these costs as a first step.

5.3 KPI Analysis

One method of determining the quality and usefulness of a metric is known as S.M.A.R.T [Staff, 2012]:

- S – Specific: a metric should be clear, unambiguous and easily understood.
- M – Measurable: Avoid “yes/no” metric; can be quantified and measured against other data
- A – Attainable: Setting goals based on unrealistic metrics aren’t helpful. It should be reasonable and credible under normal conditions.
- R – Realistic: They should fit within the company goals
- T – Timely: It should be achievable within the time frame given

To develop this project, it was necessary to use the interview method to collect and work out the requirements. Subsequently, an analysis was made in order to select some relevant ones within the timeline available. In Table 4 it is possible to see which ones were chosen.

Table 4 – Proposed Requirement

Requirements Nrº	Description
1.	Identify the company's activity by geographic area
2.	Identify which roads that are frequently used by kms and/or time and/or type of vehicles
3.	Know if the driver drives mostly at night or day; how it influences on fuel consumption and distance travelled
4.	Identify the direction of the road axes
5.	Identify the vehicles that go further without supplying fuel
6.	Identify the drivers that use vehicles for personal use

In what relates to the first requirement, it requires geographic data. So, a dimension called Geography is necessary to be created using the Kimball Methodology [Kimball and Ross, 2013] in order to provide drill down and rolling up operations with data. Through research it has been possible to find templates that relate to the desired draft [Mann, 2009], which resulted in the creation of the fields described in Table 5.

Table 5 – Fields of DimGeography

Name	Description
geography_key	A primary key for referencing
continent	Abbreviation of continent
country	Full country name
country_abbrv	Abbreviation of country (ISO-3166-3)
state	Indicates which state city belongs (applied only in some countries)
county	County name
City	City name
district	District name
cp4	Postal Code 1ª Part
cp3	Postal Code 2ª Part

The fields presented in Table 5 allow drill down through data from continent to postal code (identified by fields cp4 and cp3) allowing for a wide range of option to select (e.g. aggregate data by area). In section 6 is described in detail how information is acquired and why postal codes are used with this standard.

Since it also refers to company activity, it is necessary to design a table which contains that information stored. Therefore some fields, which Table 6 is based on, were implemented from original data source table, while others were added based on logic and on the information encountered [Williams, B., 2010].

Table 6 – Fields of DimCompany

Name	Description
company_key	A primary key for referencing
company_old_key	Old key from source system
name_company	Company name
abbrev	Abbreviated company name
company_society	Fullpubliccompanyname
address	The address of company headquarters
phone_private	Inform the private phone number
phone_public	Inform the public phone number
nif	References the contribution of number
email	Indicates the private email
site	Indicates the company website
latitude	Stores the latitude coordinate
longitude	Stores the longitude coordinate
creation_date	Record date of creation
expire_date	Represent when record Expire
active	Inform if current record is active: 1- True

Table 6 is only an example of some fields that can be encountered in a dimension of this type, which can grow to include fax phone number, name of a person that can be contacted at the company, the person's email address, etc. [Savidge, 2000]. As additional information, and not present on the source system it is added latitude and longitude fields to pinpoint the company as POI (Point of Interest) on the map; and the last three fields (creation_date, expire_date, active) implement the Slow Changing Dimension (SCD) mechanism (section 3.6) of type 2.

The second question focuses how information about roads should be organized and catalogued. During research a thesis came up [Hermannsson, 2005] that uses the approach of aggregating roads into segments to identify possible events that may occur during the driving stage (like speeding, accident, traffic, etc.). Some fields are based on that thesis and have been adapted using the information obtained from generic API Routing.

Table 7 – Fields of DimRoad

Name	Description
road_key	A primary key for referencing
geography_key	A surrogate key for referencing <i>DimGeography</i>
name_road	Street Name
segment_nokia	An identification given by Here Maps
direction	One-way: 1- direction to on side; 2- opposite direction; 3 – Both ways
category_nokia	Road classification according Here Maps
category_road	Road classification according MapQuest
geometry_nokia	Coordinates matrix shape from Here Maps
length_nokia	Road segment length

band_velocity	Speed band recommends by nokia
road_geometry	Conversion <i>geometry_nokia</i> points type to geometry

To build Table 7 it was necessary to do some research about the solutions offered by the market. The study focused on Google Maps API [Developers, 2015] and Map Quest [MapQuest, 2015] and Here Maps [HereMaps, 2015b] to understand the feasibility of using each one. Since it was agreed with the organization that priority would be given to the use of the tools already implemented if possible, it was then decided to use Here Maps.

The field geography key stores can be considered a complement to Table 5 (DimGeography), since a road is associated with a district or city, country, and continent. In relation to the nokia segment, it gives a string of characters which subsequently allows reconstructing the route of vehicle using Enterprise Route Version of Here Maps. The fourth requirement refers driving direction on road (Table 4) which is also given by API. The acquired road geometry gives an array of points which defines a LineString. In order to give relevance to any query made to this parameter, this field went through transformation to the geometry type supported by PostGIS.

Since the second requirement of Table 4, also involves vehicle description, it led to the construction of the vehicle table. The structure has undergone few changes during the data migration process, once the relevant fields of the table of business were already indicated in the source table. Of those that can stand out in Table 8 are vehicle old key, indicative key of original source database and the update mechanism SCD type 2 (create_date, expire_date, active).

Table 8 – Fields of DimVehicle

Name	Description
vehicle_key	A primary key for referencing
vehicle_old_key	Primary key in the original data source
registration	Identification of vehicle's registration
brand	The brand associated with the vehicle
model	The corresponding model
typology	Can vary from heavy passenger heavy, light goods, light passenger, etc.
classification	Identify type of vehicle (e.g. truck or car)
year	The year of manufacturing
equipment	Type of GPS equipment has installed
locked	Identification if is locked or not
fuel_type	Inform about the type of fuel
fuel_tank	Capacity of fuel tank
fuel_consumption	Indicates the average fuel consumption
engine_tecnology	Description of the agreement with the emissions standard
km_ini_activity	Indicates how many Km had during acquisition
map_identification	Identification: 0 - registration, 1- designation
designation	Personalized value that references the

	vehicle
sensors	Identifies whether the vehicle has cooled ark (1-True, 0-False)
creation_date	Record date of creation
expire_date	Represent when record Expire
active	Inform if current record is active: 1- True

Table 8 identifies some of the possible attributes that can be included in this table. For example, fields like type transmission, horsepower, dimensions, weight, among others, can be added. This information can be easily obtained with web API free: Edmunds API [Edmunds, 2015], Fuel Economy [Energy, 2015] or Kee Resources [KeeResources, 2015]. To use these free service it is necessary to get a valid key through an electronic registry, and use it with limit request constraints. It is for this reason that most databases with this data, are paid.

The third request evaluates data and time, driver and fuel consumption, where part of the problem was deciding which approach to take. According to Kimball [Kimball and Ross, 2013] the date and time should be divided into different tables. Thus, the following structure concerning Date is provided by the author (Table 9).

Table 9 – Fields of DimDate

Name	Description
date_key	A primary smart key for referencing year,month,day
full_date	The date in format dd-MM-YYYY
day_of_week	Day of week in numeric format
day_num_in_month	Indicates which is the day of the month
day_num_overall	Count of days from the beginning of calendar
day_name	The day name
day_abbrev	The abbreviation of week name
weekday_flag	True or false if is weekday
week_num_in_year	The week number from the beginning of calendar
week_num_overall	The week number from the beginning of calendar
week_begin_date	Identifies the first week day on date type
week_begin_date_key	A smart key for referencing year, month, day of 'week_begin_date'
month	The month in numeric format
month_num_overall	Month in numeric format from the beginning of calendar
month_name	The name of month
month_abbrev	The abbreviation of the name of month
quarter	The quarter in numeric format
year	The year
yearmo	The year and month together

fiscal_month	Current fiscal month
fiscal_quarter	Indicates the fiscal quarter
fiscal_year	Indicates the fiscal year
last_day_in_month_flag	A flag that signal if is last day in month
same_day_year_ago	Identify which day was a year ago

The Time dimension (Table 10) was built with default parameters (hour, minutes, and seconds) and other relevant attribute. Therefore, it is possible to answer the question whether the driver is driving in the morning or in the evening.

Table 10 – Fields of DimTime

Name	Description
time_key	A primary smart key for referencing hours, minutes , seconds
hours24	The hour that ranges from 0 to 23
hours12	The hour that ranges from 0 to 11
minutes	The minutes
seconds	The seconds of the hour
am_pm	The variable that identify AM or PM hours
time_value	Time value of complete hour

Another dimension necessary to respond to the third requirement is driver information. The proposed structure is based on AdventureWorks database and employee table [Microsoft, 2008]. Only the necessary fields needed to the business area are presented. The dimension is shown in Table 11.

Table 11 – Fields of DimDriver

Name	Description
driver_key	A primary key for referencing
driver_old_key	Primary key in the original data source
Licence	Driving License
Namep	Driver first name
Namef	Driver last name
Phone	Personal phone number
identification_code	Represent a code RFID
name_navigator	Represents a chosen driver name
creation_date	Record date of creation
expire_date	Represent when record Expire
Active	Inform if current record is active: 1- True

The current Table 12 identifies a service, which is usually assigned to a driver. The table is merely an example adapted from data mart [Williams, B., 2010] about products, orders and deliveries; to track trips assigned to drivers/vehicles.

Table 12 – Fields of DimService

Name	Description
service_key	A primary key for referencing
service_old_key	Primary key in the original data source
date_plan_begin	Date planned for service start
Date_plan_end	Date planned for service end
Date_actual_begin	The actual date service starts
Date_actual_end	The actual date service ended
source	Informs where the route starts
destination	Informs the last point of service

To meet other requirements as distance, fuel economy or travel time, these were considered measures. So, the philosophy of Kimball [Kimball and Ross, 2013] state these measures should be stored in fact tables in order to do mathematical operations, and therefore the keys associated.

In any situation of information-gathering, is essential to identify the Five Ws [Buttry, 2011]. They constitute a formula for getting the complete story on a subject. According to the principle of the Five Ws, a report can only be considered complete if it answers these questions starting with:

- **Who** did that?
- **What** happened?
- **Where** did it take place?
- **When** did it take place?
- **Why** did that happen?

These five questions summarize the key points or the identification of the previously mentioned tables. For example, **who**, represents a driver (driver dimension), **what**, the driving situation normal (vehicle dimension), **where**, identified by location (geography and road dimension), **when**, associated date and time (date and time dimension) and **why**, is on service? (service dimension). This logic of questions is one of the main foundations associated with the fact table, as in this project it is shown on Table 13.

Table 13 – Fields of FactGPS

Name	Description
date_utc_key	A surrogate key from <i>DimDate</i> referencing UTC date
time_utc_key	A surrogate key from <i>DimTime</i> referencing UTC time
date_local_key	A surrogate key from <i>DimDate</i> referencing Local date
time_local_key	A surrogate key from <i>DimTime</i> referencing Local time
geography_key	A surrogate key from <i>DimGeography</i>

road_key	A surrogate key from <i>DimRoad</i>
vehicle_key	A surrogate key from <i>DimVehicle</i>
company_key	A surrogate key from <i>DimCompany</i>
driver_key	A surrogate key from <i>DimDriver</i>
service_key	A surrogate key from <i>DimService</i>
Latitude	Specify the coordinate latitude
Longitude	Specify the coordinate longitude
Direction	Value acquired from OBD equipment
Elevation	Value given by MapQuest API
Velocity	Value acquired from OBD equipment (instantaneous)
Temperature	Engine Temperature
Rpm	Value acquired from OBD equipment about engine RPM (Revolutions per minute)
Runtime	Run time since engine started
Throttle	Hand gas lever (Range 0 to 100 %)
fuel_level	Level of current fuel
fuel_rate	Fuel Economy
fuel_economy	Instantaneous Fuel Economy
km_acc	Accumulated km since the vehicle was recorded in the database
has_ignition	Signals if the vehicle is turned on
has_velocity	Signals if the vehicle has velocity
is_driving	Signals if the vehicle is being driven
is_idle	Signals if the vehicle is idle
Is_service	Signals if the vehicle is being driven with assigned service
fuel_consumption_est	Parameter calculated

The fields date and time existing on Table 13 are duplicated, since the data source system deals with multiple time zones, Kimball says that it is advisable to implement this approach, such as illustrated in the book [Kimball and Ross, 2013] in an airport example. Since it is also necessary to store the coordinates GPS on a table that will have a high growth due to the cadence data to the second, Kimball speaks that the largest tables in this system should be the facts tables. For this reason, the latitude and longitude fields are chosen to be stored in.

The gps coordinates, direction, velocity, temperature, RPM, throttle, run time, fuel level, fuel rate and fuel economy are some of the possible fields which can be acquired by the OBD equipment installed in the vehicle. An OBD (On-Board diagnostics) is an automotive term that often refers to a vehicle's self-diagnostic and reporting capability. Modern OBD implementations use a standardized digital communications port to provide real-time data in addition to a standardized series of diagnostic trouble codes, or DTCs. Essentially it can detect problems long before the driver is able to notice any symptoms, such as low-performance, low-fuel economy, and heavy emissions, or before the Check Engine or Malfunction light comes on [Bolduc, 2014].

Since almost every car and truck have OBD systems installed, the problem lies on older OBD systems (which had their own set of standards) or caterpillar vehicles which do not have any

connections. So, it is not possible to store all the required parameters in Table 13. As one of the requirements surrounding the development of this project was to estimate the calculation of fuel consumption after a trip, a possibly viable option would be mainly through gps coordinates and other calculated fields. A work studied was *"Mining Geographic Data for Fuel Consumption Estimation"* [Ribeiro, Rodrigues and Aguiar, 2013] which focus on estimating the instantaneous fuel consumption from the smartphone's GPS data alone, using the OBD through speed, acceleration and steepness as predictor variables to train polynomial models with and without cross-product terms. The report gives a good perspective good results with an average residual standard deviation of 1.58 l/100km for average consumption on 1min/intervals, however the proposed solution only focus on specific car models. Other solution consulted was *"Indirect Instantaneous Car-Fuel Consumption Measurements"* [Skog and Handel, 2014] which again focus GPS coordinates and measures by the car's on-board diagnostics (OBD) data bus, presenting a more rigorous mathematical model involving variables such as mass density of air, Roll resistance force, Tractive force, Air drag force, etc. Same problem arises once again since it only emphasizes attention on cars and not contemplates trucks.

In order to understand which variables influence the fuel consumption according to the driving mode, the publication *"Evaluating Eco-driving Advice using GPS and CANBus data"* [Jakobsen, Mouritsen and Torp, 2013] was studied. Besides making reference to fuel consumption, it focuses attention on study eco-drive styles and seeks to provide reference to how fuel efficient consumption could improve.

To sum up, it is difficult to find a generic formula that could allow comparison between vehicles due to diversity of driving type (like driving fast, break very often, etc.), vehicle characteristics (weight, aerodynamics, tires, etc.) and environment conditions (winding road, poor condition of pavement, etc.). From different studies and approaches to deal with the problem, it was used a formula retrieved from *"EcoMark: Evaluating Models of Vehicular Environmental Impact"* [Guo, Ma, Yang, Jensen, C.S., et al., 2012], which calculates the instantaneous consumption given a specific moment, using vehicle speed, acceleration and road grade; value stored in Table 13 as *fuel_consumption_est*.

In the original data source, vehicle speed is a required parameter, so assuming this to be true it is possible to calculate the parameter of acceleration without problem. However road grade is a parameter not obtained from any source, which it is essential to make the formula [Guo, Ma, Yang, Jensen, C., et al., 2012] viable. Therefore the initial approach was to use HereMaps [HereMaps, 2015b] but since it does not provide, until the date, the necessary feature, it was decided to choose MapQuest [MapQuest, 2015].

As for the fields it has ignition, has velocity, is driving, is idle, is service they are true or false values. It follows the structure of *"An Advanced Data Warehouse for Integrating Large Sets of GPS Data"* [Andersen et al., 2014], which is an implemented solution in Denmark, extended with the project available at www.daisy.aau.dk/its which estimates fuel consumption in a route.

5.4 KPI Exploration

Thereupon this section focuses on describing some of the common metrics analysis in this thesis. The most suitable approach of KPI analysis is to use a comprehensive Table 14 illustration for each KPI. Each one as several attributes; they are name, measure and description.

Table 14 – KPI card table explanation

Attribute	Content
Name	Name is a basic description of the KPI, which demonstrate characteristics related to the measure.
Measure (Expression)	A measure is a calculation method which transform abstract indicators into numerical form. Normally they are apply mathematical formulas to measure KPIs and variables of the formula.
Description	Detailed description for the KPI is to explain the reason for the importance of the selected KPI, clarify the environment of measuring and illustrate the key role of the entire enterprise

5.4.1 Trip Time & Distance

The travelled millage is one typical measure in which fleet managers are typically interested. It provides a monitoring about use of vehicle, illustrating events that may occur during the driving stage. It is possible to see the metric on Table 15 .

Table 15 – KPI card of Trip Distance

Attribute	Content
Name	Trip Distance (Km)
Measure (Expression)	$Km_{end_vehicle_trip} - Km_{begin_vehicle_trip}$
Description	Calculates the total travelled mileage that the vehicle drive through a trip, which are usually assigned (each day normally).

Since a vehicle moves over time, it is also important to detect time events. So, this is also a good metric to be stored. Here a problem can be approached in different ways, like understand how long the trip takes, idle times, the duration of each stop, etc. To simplify, it calculates the total time of an assigned trip take, as it is given in Table 16, by each day. Since all information is stored, is possible to deepening the research like idle time.

Table 16 – KPI card of Time Duration

Attribute	Content
Name	Trip Duration (Hours)
Measure (Expression)	$Time_end_vehicle_trip - Time_begin_vehicle_trip$
Description	Calculates the duration of trip with stop time and driving time, by each day.

5.4.2 Idle Time Cases

When calculating idle time, the most important point to understand is how a trip is defined (Table 17). A trip begins when the vehicle starts moving and ends when the vehicle starts moving again after a stop. A stop is recorded when the vehicle ignition is turned off, or when the vehicle has a speed of less than 5 km/h for more than 200 seconds. Any idling within a trip is associated to that trip [Loong, 2014].

Understanding these conditions and having a strategy how to handle idle time will effectively lead to accurate results. Therefore, some of the typical problems, and what the proposed solution to perform the calculation is, are set out below.

Table 17 – KPI card of Idle Time

Attribute	Content
Name	Total Idle Duration (Hours)
Measure (Expression)	$SUM(Idle\ Time)$
Description	Calculate the sum of total stop idle time of a vehicle, by each day

5.4.2.1 Case 1

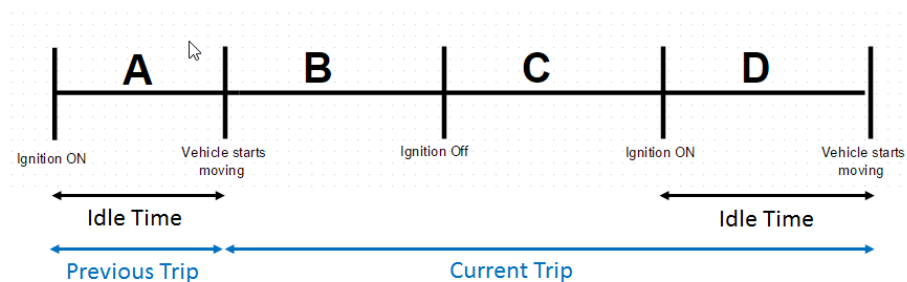


Figure 14 – Idle Time Case 1

In this example Figure 14, let's assume time A and D are times spent idling. Therefore,

Previous trip idle time = A

Current trip = B + C + D

Current trip idle time = D

5.4.2.2 Case 2

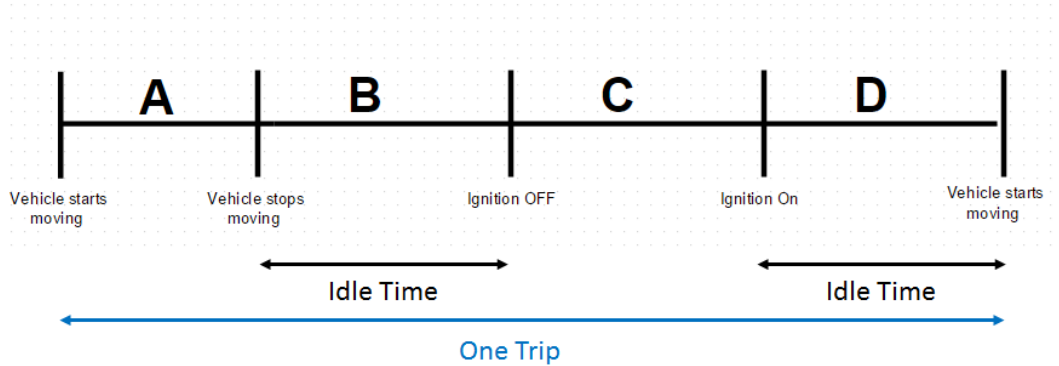


Figure 15 – Idle Time Case 2

In this example Figure 15, let's assume time B and D are times spent idling. Therefore,

Current trip = A + B + C + D

Current trip idle time = B + D

5.4.2.3 Case 3

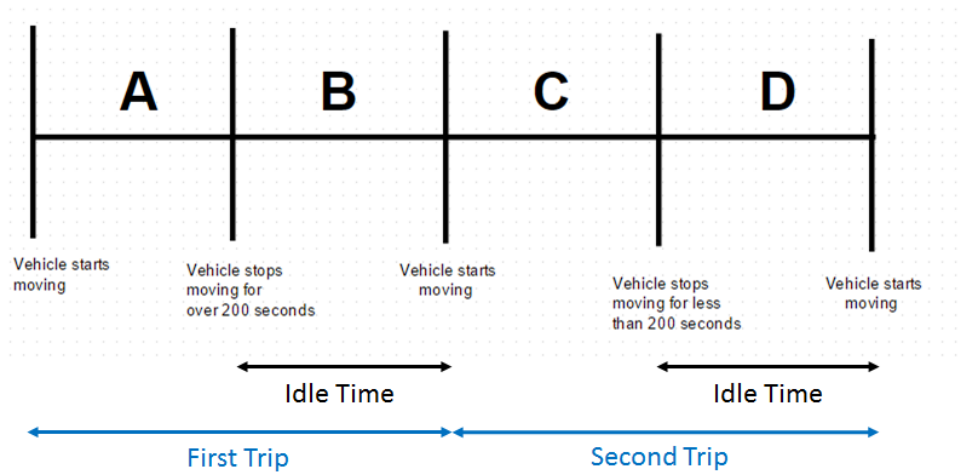


Figure 16 – Idle Time Case 3

In this example Figure 16, A+B is the first trip, and C+D is the second trip. The idle times are B and D where B is attributed to the first trip, and D is attributed to the second trip.

5.4.2.4 Case 4

If there is a gap in GPS data for more than 200 seconds (usually caused by the device being unplugged while idling), the software does not record idle time during this period, as there is no way to confirm the vehicle was idling during that time.

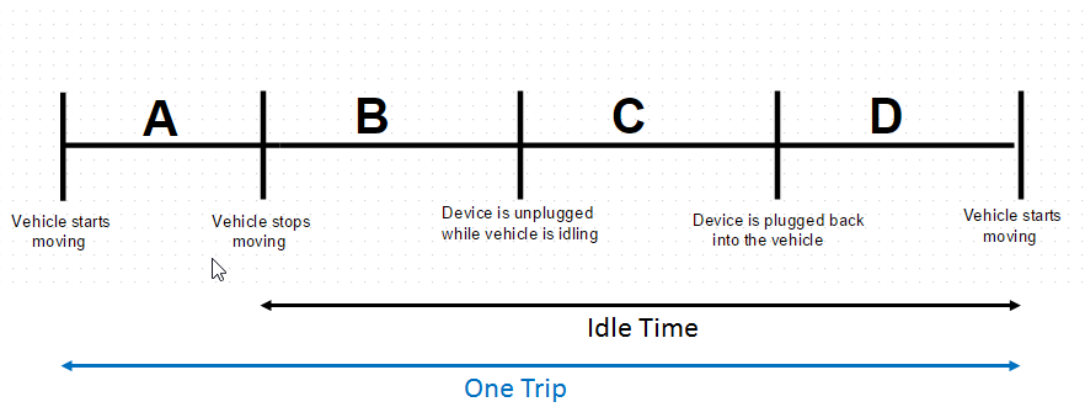


Figure 17 – Idle Time Case 4

In this example Figure 17, let's assume time B, C and D are times spent idling. Therefore,

Current trip = A + B + C + D

Current trip idle time = B + D (C does not count towards idling, because the device was unplugged)

5.4.3 Trip STOP Time

The KPI of Table 18 helps to understand if a specific driver usually takes more time than necessary to fulfil a delivery of a cargo. (Spends more time stopped than driving, compared to other drivers that do same job).

Table 18 – KPI card of Stop Time

Attribute	Content
Name	Total Stop Duration (Hours)
Measure (Expression)	$SUM(Stop\ Time)$
Description	Calculate the total stop idle time of a vehicle, by day

By deep through this KPI is able to carry out a more detailed analysis by identifying the places where drivers often rest from work and then identify those that meet the standards recommended rest. Moreover, it can be considered as a parameter to be added in the drivers rank.

5.4.4 Fuel Consumption

Nowadays the current price of fuel weighs more and more on business spending and therefore they seeks strategies to balance and reduce them (Table 19). One tactic is to improve travelled distance with same amount of fuel.

Since it is difficult to measure fuel consumption of a vehicle which does not contain devices on board, it is necessary to use mathematical equations to obtain results. The problem is explored with more detail in a following section of this dissertation (Equation 1 on page 73).

Table 19 – KPI card Fuel Consumption

Attribute	Content
Name	Total Fuel Consumption (l/h)
Measure (Expression)	$SUM(Fuel\ Consumption)$
Description	Calculate the sum of fuel consumption. Data should be aggregate by day.

6 System Implementation

This chapter is a continuation of the content described in chapter 5. Here a business intelligence system is implemented following the generic steps of any BI application [Buys, 2015], shown by Figure 18. The different dimension tables construction process is approached, as well as the fact table, which ultimately will lead to the construction of a dashboard to show the KPIs.

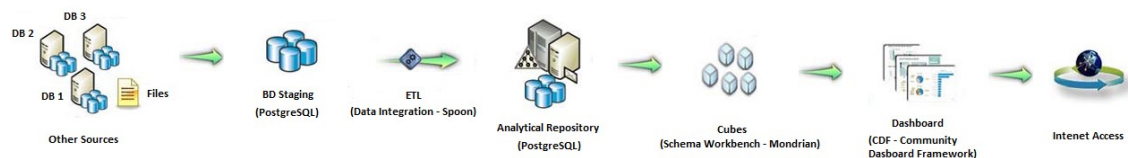


Figure 18 – Solution Data Flow

Since the Pentaho community edition offers a full stack of tools to develop a BI project, Pentaho Data Integration was selected to develop ETL processes, Schema workbench to build the cube and run it on the Pentaho Server (where it is the OLAP server) and CTools to construct a Dashboard. Since source system tables are stored in a Postgres database, the dimensional model creates a DataMart that was also developed to work with the same database server.

To start the project was required first to setup development environment, such as configuration and installation of Pentaho web server, PostgreSQL database server, Pentaho Data Integration and Schema workbench. Then, it was necessary to gather all relevant data from various sources and load them into *DB Staging*. From here, ETL process is implemented and the data are stored in fact tables and dimensional tables (stage *Analytical Repository*). According to the multidimensional analysis strategy of Olap [Kimball and Ross, 2013], the third step is create Mondrian Schema by using XML, and then commit them into Pentaho Web Sever. To finish, the last step was developing a structure to show data in a graphic interface on the web, the Dashboard.

6.1 ETL Implementation

The detailed ETL implementation on Pentaho Data Integration is explained in this section. It was decided that extraction and loading of data from the operational database to the data mart would be done with one transformation per dimension table. In order to understand each stage of the ETL relative to several dimensions, the information is divided into topic extraction, transformation and loading.

6.1.1 Configuration Connection

The first thing to perform when starting the program is to create and set up the connection with the database. In this project, it was necessary to connect to the PostgreSQL database containing several tables where data is stored. As shown in Figure 19, on its right side, it displays the procedure to following order to bring up the setup wizard.

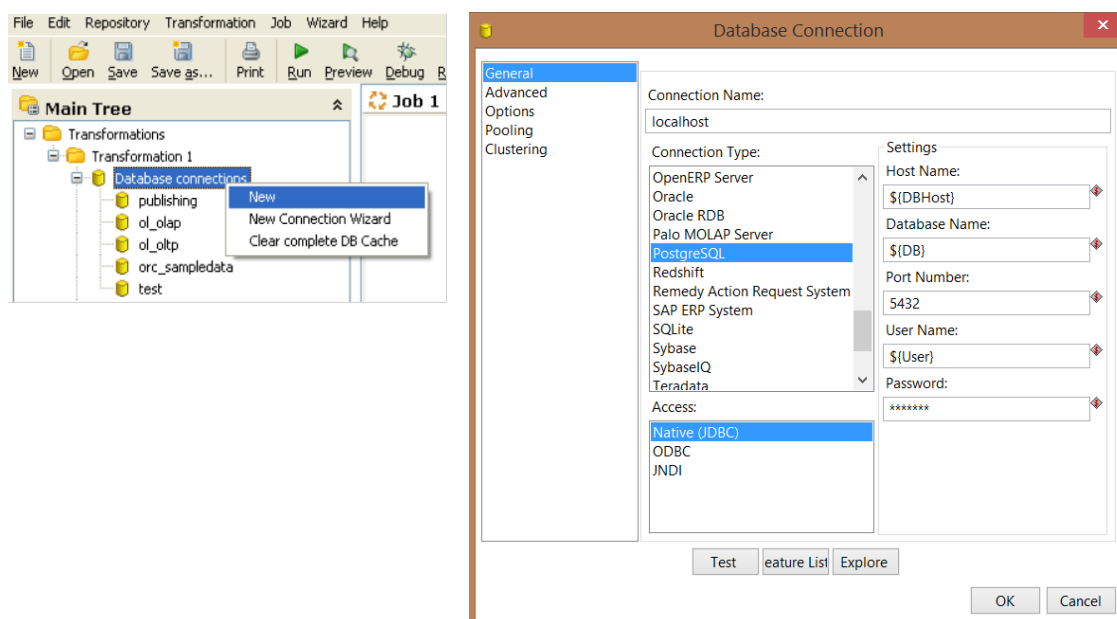


Figure 19 – Connection Setup: steps to follow (left), wizard window (right)

To configure the connection (Figure 19) it is prerequisite to identify the connection name since it uniquely identifies a connection across transformations and jobs. Then, in Connection Type, is necessary to select the type of database which are connecting to (Oracle, PostgreSQL, MySQL, etc.). By default, the drives folder is empty, which means that during installation of program, it is necessary to download the appropriate driver from the database manufacturer's website. Finally, in Settings, the hostname, database name, Port, username and password are introduced.

For safety reasons, the settings of PDI (Pentaho Data Integration) are stored in an external file. When PDI is executed, loads the configuration into variables previously set as shown in Figure 19. For example, the name of the database is stored in the variable `${DB}`.

6.1.2 Extraction Process

The PDI, referred on page 30, allows extracting information from a variety of diverse data including all popular structured, unstructured and semi-structured data sources. Some examples include standard relation databases (Oracle, DB2, MySQL, SQL Server); Hadoop, Cloudera, HortonWorks, MapR; NoSQL databases (MongoDB, Cassandra, HBase); Analytic databases (Vertica, Greenplum, Teradata); cloud-based and SaaS applications (Salesforce, Amazon Web Services); Files, XML, flat files and web service APIs [Pentaho PDI, 2015].

The component used to perform the data extraction process from the various tables in the original system, is called *Table Input*. On Figure 20 the visual design is presented on the left side and the corresponding configurations available of the component on the right side. It is advisable to change the step name to a reference which meaning is easily understood. In the case when the name is ambiguous and there are multiple data extraction sources, it is difficult to understand the origin of the data and therefore the change parameters. The connection menu displays all connections available according to Figure 19 settings, allowing selecting which most fit. Next, a SQL statement is given by default in the text area, where the custom SQL query is written which connects with single or multiple tables of the database.

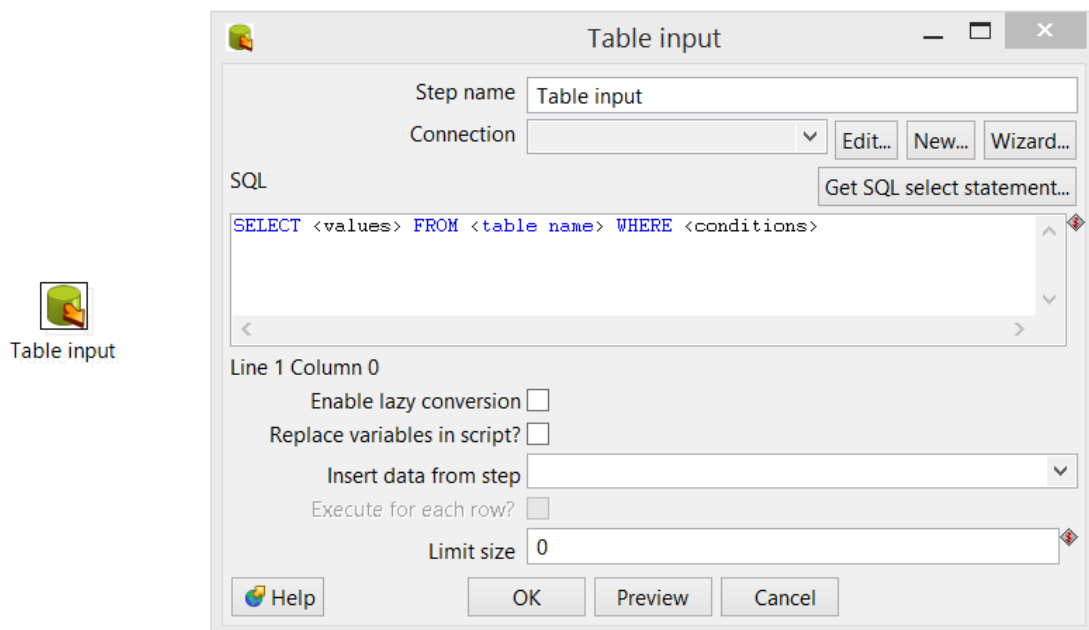


Figure 20 – *Table Input* Component PDI: visual design (left), menu settings (right)

Another present feature in Figure 20 is considered advanced. By enabling the option of lazy conversion avoids unnecessary data type conversions and can result in a significant performance improvement. In case of use the *replace variables in script*, it indicates that another components connected and sending values. Thus implies selecting a step name from the box of *Insert data from step*. *Execute for each row*, as names states, enables to perform

the data insert for each individual row. Regarding *Limit size*, sets the number of lines that is read from the database; zero (0) means read all lines [PentahoWiki, 2010].

As it is shown in Figure 20, data extraction is performed through an inquiry to the database, through the use of SQL (Structured Query Language). SQL is the declarative query language standard for relational database, where many of the original features were inspired by the relational algebra.

In every dimension previously cited on chapter 5, component of Figure 20 was used to extract information, except date and time. For example, in Code 2 a simple example of the query used to extract information from a driver table is shown, since for privacy reasons, it is not possible to list all the queries.

```
SELECT*
FROM   condutor
```

Code 2 – SQL Statement Driver table

Data extraction intends to select all the values of the driver table. SQL enables to read all using asterisk character (*).

To create Date Dimension, a component called *Text file input*, was used. As is a table founded in all BI projects, Kimball group [Kimball and Ross, 2013] offers a spread sheet to be loaded into systems. To do so, it is then used this component (Figure 21) that performs the file reading.

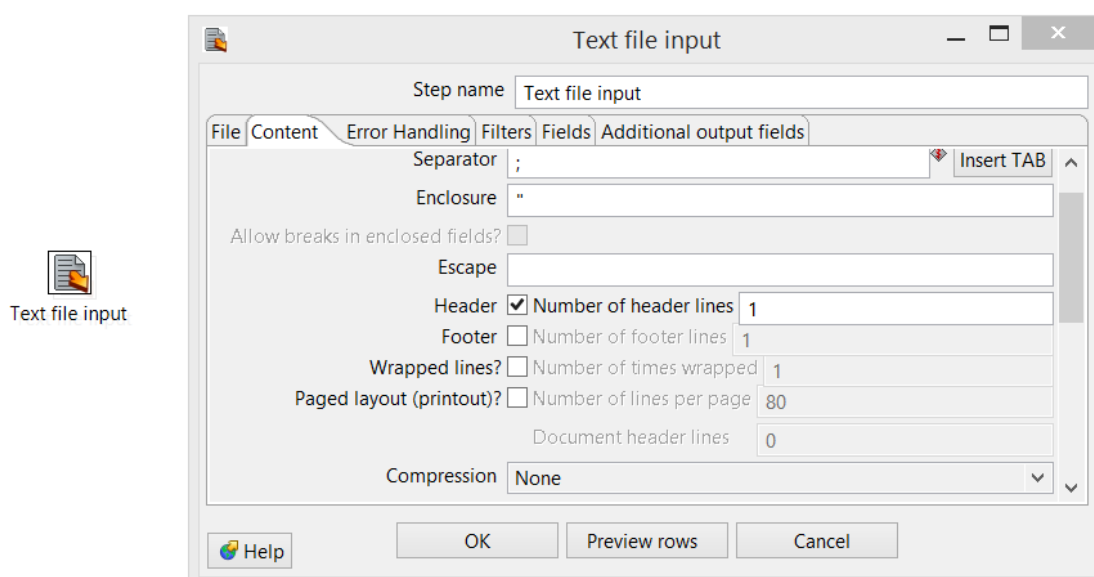


Figure 21 – Text File Input: visual design (left), menu settings (right)

The *Text File Input* step is used to read data from a variety of different text-file types. The most commonly used formats include Comma Separated Values (CSV files) generated by spreadsheets and fixed width flat files. The Text File Input step provides the ability to specify a list of files to read, or a list of directories with wild cards in the form of regular expressions. In addition, it can accept filenames from a previous step making filename handling even more generic [Pentaho PDI, 2015].

Nonetheless it is important to understand how data should be extracted. The two principal methods are a static method which capture a snapshot of all source data at a point in time, and incremental extract that captures only the changes that have occurred in the source data since last capture. The last method was chosen to be used with log records.

6.1.3 Transformation Process

This sub-section address the transformation of stream processes implemented through table's dimensions before being loaded. A generic view is presented with Table 20, based on structure proposed by Kimball Group [Kimball and Ross, 2013] as part of Logic Data Map. Normally this stage can also be called as Data Cleaning and Conforming, since these are the main operations to carry out.

Table 20 – Overview Essential Transformation Process

Target Table	Transform Process
DimDate	- Convert datatype from String to Integer of field DateKey
DimTime	- Generate Rows (Hours, Minutes, Seconds) - Generate Primary Key
DimDriver	- Validate Driver's Licence - Validate Driver's Name - Validate Driver's Phone
DimCompany	- Validate Address Company - Validate Phone - Validate Email
DimVehicle	- Validate Year Model
DimService	- Generate Primary Key
DimGeography	- Retrieved with API
DimRoad	- Retrieved with API

In the next examples the various dimensions are displayed (according to Table 20), in order to explain the process of transformation that the teach table has undergone before being loaded, figures and tables are also shown, with the purpose of explaining the use of each component presented.

6.1.3.1 Date Dimension

As mentioned before, Date Dimension loads data from a spreadsheet. The full ETL process is given by Figure 22. The transformations applied are described in Table 21.

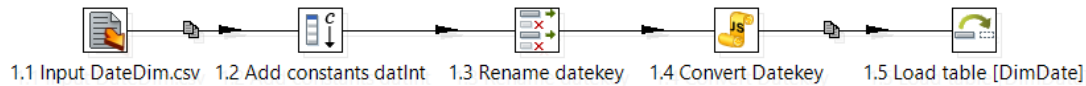


Figure 22 – ETL of Date Dimension

Table 21 – Description ETL process Date Dimension

Step	Operation Type	Description
1.1 Input DateDim.csv	CSV Input	This step extract data from specific file with format type CSV. Here all fields are specified with data type, format (integer, year, etc.), length, precision, etc.
1.2 Add constants datInt	Add constant values	This step add two fields with constants (datInt and datString, integer and string respectively).
1.3 Rename datekey	Select / Rename Values	Rename value of main key <i>data</i> to <i>datekey</i> .
1.4 Convert Datekey	Modified Java Script Value	Modify value using formula of field <i>datString</i> : <code>var datString = date2str(datekey, 'yyyyMMdd');</code>
1.5 Load Table [DimDate]	Insert/Update	This step is loading which is to load the data stream to matched field DimDate table

This stream (Figure 22 and Table 21) introduces the ETL process of date dimension. The main problem is due to Pentaho Data Integration was having problems to read dates in format 'yyyyMMdd' and so, an alternative was needed to overcome this difficulty.

Another option to generate the same fields as in the spreadsheet was to include in ETL process an operation type *Calculator*. This component given a date, allows the use of formulas like "Day of week of Date A", "Day of year of Date A", "Week of year of Date A", etc. to generate *day_in_week*, *day_in_year*, *week_in_year* respectively.

6.1.3.2 Time Dimension

When is installed PDI, the software contains samples illustrating the use of the different components present in the software. In one case, the program exemplifies the construction of the time dimension table which generates column hour, minutes and seconds. Since it was intended to add some extra columns, it was decided to use this transformation and adapt it to fulfil the requirements, as it is shown Figure 23.

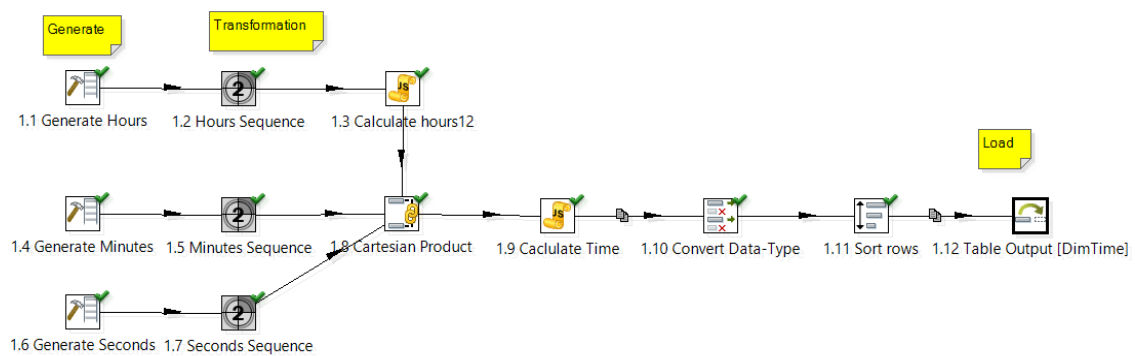


Figure 23 – ETL of Time Dimension

In Step 1.3 the field called `hours12` verifies if current hour value belongs to the range of 0 to 11. Since this field is confusing about differentiating day and night, the am/pm field was implemented to check if it is dawn (am) or afternoon (pm). The component 1.9 generates the primary key, with format 010000 corresponding to hour "01:00:00". It is considered as semantic key [Kimball and Ross, 2013] because the values generated, has a clear meaning for the user.

6.1.3.3 Driver Dimension

The cleaning and conformation of data in this dimension required a bit of work since the original data considers drivers with different nationalities, implying different standards. Since most of the data is associated to Portuguese clients, it was decided to use only a Portuguese standard to deal with validations, which served as a model for future improvements. Figure 24 shows the implemented steps.

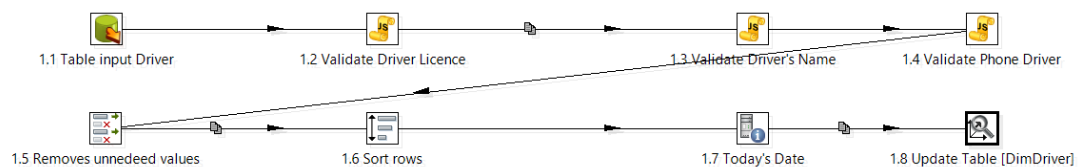


Figure 24 – ETL of Driver Dimension

As it can be seen in Figure 24, there are three components that by default are called *Modified Java Script Value* which could be joined into one component. This approach is only intended to clarify the process of each validation. The step 1.2 uses *regular expressions* (a sequence of characters that define a search pattern, mainly for use in pattern matching with strings, or string matching), to identify if a licence is valid according to the law [Decreto-Lei nº 37/2014 de 14 de Março do Ministério da Economia, 2014].

Step 1.3 also applies pattern matching to clean numbers, accepts only one name and uppercases the first letter. With step 1.4, it validates driver's phone using the pattern `(/(\d{3}[/-]*){3}/g)` to find numbers like 999999999, 999 999 999, 999-999-999, 999/999/999 (remove letters mixed with numbers and accept only nine digits). Then a

standard Portuguese indicative (00351) is added. Step 1.7 is used to obtain the current system data and then insert or update it depending of the data (step 1.8), using the slowly changing the dimension mechanism (SCD) type 2. In case of an unsuccessful validation of fields, they are discarded and replaced with a predefined value.

6.1.3.4 Company Dimension

The transformation from Figure 25 involves mainly the cleaning and conforming of the address from different business/clients, and merged with extra information in order to complete it. This flat file (step 1.2) include latitude, longitude as point of interest to display on the map, include full public name of business, or include web site reference, being hand made by consulting each business web site.

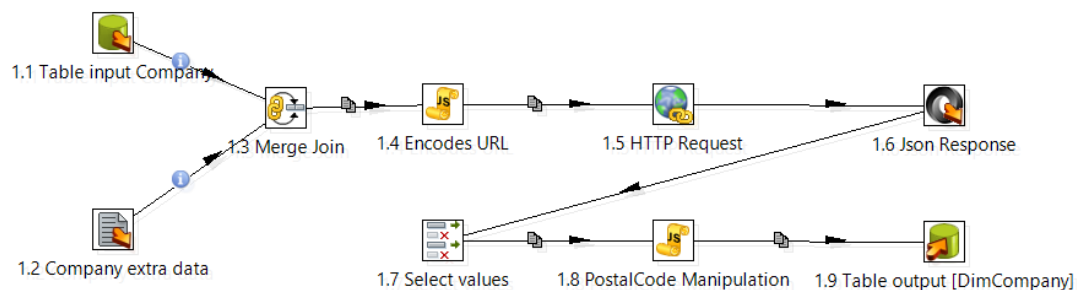


Figure 25 – ETL of Company Dimension

The Step 1.4 uses a component “Modified Java Script Value” to encode URL of each row using the reverse geocode API based on the values of latitude and longitude. Step 1.8 divides the Portuguese Postal Code into two parts (separated by delimiter “-”), since it is designed to be used in this way, because over 90% clients are Portuguese [GeoPostcodes, 2015].

Reverse Geocoder API

This web API reverse geocodes retrieves a street address or administrative area information corresponding to a given geo-coordinates. By this way is possible to create a standard concerning to the address before being stored.

To make a request is only necessary any HTTP client application using the parameters[HereMaps, 2015a]:

- app_id → represents the Application ID
- app_code → represents the Authentication Token
- gen → with *gen*>7 the radius is not ignored. (is optional feature)
- prox → latitude, longitude, radius
- mode → type of mode which retrieve information

Next, is presented an example of how to make a valid request to web service given latitude=41.14117 and longitude =-8.60891 with radius 10 meters.

```
http://reverse.geocoder.cit.api.here.com/6.2/reversegeocode.xml?app_id=D
emoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
&gen=8
&prox=41.14117,-8.60891,10
&mode=retrieveAddresses
```

Code 3 – Reverse Geocode Request

The current result obtained from request Code 3 is displayed in JSON format by Code 4:

```
<ns2:Search>
  <Response>
    <MetaInfo></MetaInfo>
    <View xsi:type="ns2:SearchResultsViewType">
      <ViewId>0</ViewId>
      <Result>
        <Relevance>1.0</Relevance>
        <Distance>-76.0</Distance>
        <Direction>16.8</Direction>
        <MatchLevel>district</MatchLevel>
        <MatchQuality></MatchQuality>
        <Location>
          <LocationId>51M4uffH+o0zKQVdYvUbaA</LocationId>
          <LocationType>area</LocationType>
          <DisplayPosition></DisplayPosition>
          <MapView>
            <TopLeft></TopLeft>
            <BottomRight></BottomRight>
          </MapView>
          <Address>
            <TopLeft></TopLeft>
            <BottomRight></BottomRight>
          </MapView>
          <Address>
            <Label>Sé, Porto, Portugal</Label>
            <Country>PRT</Country>
            <County>Porto</County>
            <City>Porto</City>
            <District>Sé</District>
            <PostalCode>4000-098</PostalCode>
            <AdditionalData
key="CountryName">Portugal</AdditionalData>
              <AdditionalData
key="CountyName">Porto</AdditionalData>
            </Address>
            <MapReference></MapReference>
          </Location>
        </Result>
      </View>
    </Response>
  </ns2:Search>
```

Code 4 – Reverse Geocode Response

From the result from Code 4 it is possible to see information arranged by tags (markup language structures consisting of short instructions having a start and an end) whereas Country, County, City, District and Postal Code are some columns of table *DimGeography*. By separating the information like this is possible to identify geographic areas of business at the level of Postal Code (e.g. allow to search companies in Porto, by Sé district). Since this division of the addresses was not fully developed because of time constrains, information is stored separated by commas using a standard (Country, initials country , county, city, district, postal code) as it follows in an example from Code 4: Portugal, PRT, Porto, Porto, Sé,4000-098.

6.1.3.5 Vehicle Dimension

Vehicle Dimension collects and stores all information about vehicles registered on the source system. Here, like in other streams, it was required to do some validations and SQL join queries from different tables. Figure 26 shows the ETL process of vehicle dimension, and the detailed description of each step of the ETL process is listed in Table 22.

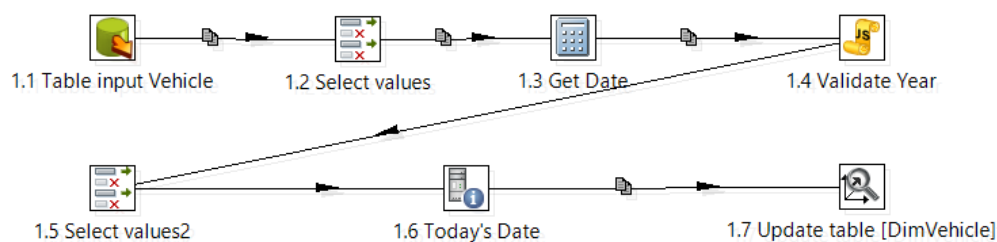


Figure 26 – ETL of Vehicle Dimension

Table 22 – Description ETL process Vehicle Dimension

Step	Operation Type	Description
1.1 Table input Vehicle	Table input	This step extracts data from table Vehicle and Brand. It uses the query: <code>SELECT * FROM vehicle v, brand m Wherev.id_brand = m.id_brand</code>
1.2 Select Values	Select / Rename Values	This step rename some column's name.
1.3 Get Data	Calculator	This step get from column date in timestamp, create a column year, month, day, hours, minutes and seconds.
1.4 Validate Year	Modified Java Script Value	This step applies validations and data-type conversions to field year which belongs Brand table.
1.5 Select Values2	Select / Rename Values	This step removes unneeded columns created during step 1.4
1.6 Today's Date	Get System Data	This steps as the name states, gets the today's date from current system.

1.7 Update table [DimVehicle]	Dimension Lookup/Update	This step loads data into DimVehicle table. Otherwise, if the current row already exists, is then updated the table.
-------------------------------	-------------------------	--

The source tables containing brand and vehicle were required to be joined in order to conform to this table dimension, so step 1.1 was needed. Step 1.3 (Get Data) retrieves the year model, and applies a standard date format (e.g. year 14 is conform to 2014 or 20014 to 2014) in step 1.4. If not present in original system, then a default value 0000 is assigned to the vehicle. Step 1.6 and step 1.7 implement the mechanism of update SCD type 2.

6.1.3.6 Service Dimension

Service Dimension represents planned trips that are assigned to the vehicle. Therefore, the initial data stream of service starts by selecting the source table (step 1.1) and then step 1.2 is used to look up and match the new company key with the old one. Figure 27 shows the ETL process which was developed.

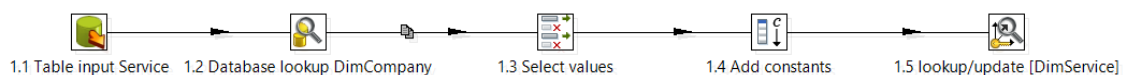


Figure 27 – ETL of Service Dimension

The menu settings of component database lookup *DimCompany* in Figure 27 are displayed in Figure 28. To lookup a value it is necessary to indicate a previous connection configured and the source table which one wants to lookup. From there, specify the table field name from source and current name field from the stream which one wants to compare (identify by area 1 in Figure 28). Lastly, a new column is created according to configurations of Figure 28 (area 2).

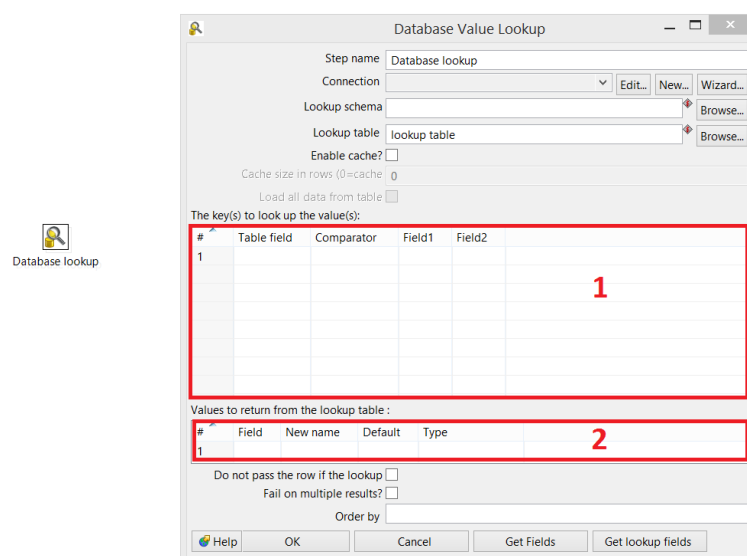


Figure 28 – Database Lookup: visual design (left), menu settings (right)

6.1.4 Fact Table

After preparing the tables dimensions comes one cornerstone of the project, the fact table. Here, the information is merged with multiple sources improving the knowledge of present data in the database, such as APIs and flat files. Given the length of the stream, it was divided into Figure 29, Figure 30, Figure 31, Figure 32 and Figure 33.

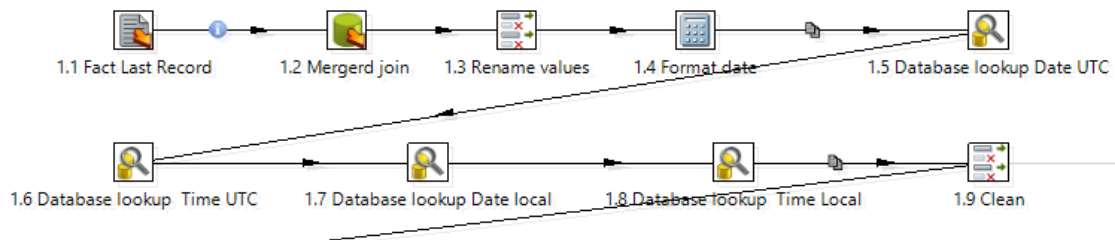


Figure 29 – Extraction and Look up tables (Fact_GPSpart 1)

Table 23 – Description ETL process Figure 29

Step	Operation Type	Description
1.1 Fact Last Record	CSV input	This step aimed to extract information from file. It verifies what the last record stored in Fact_GPS table was.
1.2 Merged Join	Table input	This step merges two tables using the query: <pre>SELECT distinct c.im, (c.hourdate::timestamp) as date_utc, (c.hourdate::timestamp) as time_utc, (c.hourdate::timestamp at time zone 'UTC') as date_local, (c.hourdate::timestamp at time zone 'UTC') as time_local, c.lat, c.lon, c.velocity, c.per, c.km, c.street, c.direc, o.* FROM coorold_2012_04 c INNER JOIN obdold_2012_04 o ON c.hourdate = o.hourdate where c.hourdate::date >= ? and c.hourdate::time >= ? order by c.hourdate, c.im asc</pre>
1.3 Rename Values	Select / Rename Values	This step rename some column's name, and display results with a predefined order.
1.4 Format Date	Calculator	This step extract year, month, day, hour, minutes and seconds from "hourdate", creating respective fields.
1.5 Database lookup Date UTC	Database Value Lookup	This step looks for in DimDate the key which match year, month and day in the stream. Value is in UTC.
1.6 Database lookup	Database Value	This step looks for in DimTime the key which

Time UTC	Lookup	match hour, minute and seconds in the stream. Value is in UTC.
1.7 Database lookup Date Local	Database Value Lookup	This step looks for in DimTime the key which match year, month and day of specific location.
1.8 Database lookup Time Local	Database Value Lookup	This step looks for in DimTime the key which match hour, minute and seconds of specific location.
1.9 Clean	Select / Rename Values	This step is aimed at removing previous dates in the timestamp.

Step 1.1 implements a text file to save the last record inserted into the database. So, the next incremental loading information starts from this point on, avoiding loading all information again. One of the downsides to deal with huge data information is how to make it easily searchable on acceptable times. To become affordable, the information was distributed by month which can be noticed by the query in Step 1.2. It is therefore here where the two major tables containing information about gps track position and information about vehicle are merged.

Considering that vehicles circulate through more than one country, it was necessary to duplicate the fields (*hourdate* in step 1.2) in order to make it searchable by local time given a specific country or by time in standard as UTC. Step 1.4 extracts the information from field's *date_utc*, *time_utc*, *date_local* and *time_local*, creating an extra field. Therefore, using this new attribute, then it is possible, with steps 1.5, 1.6, 1.7 and 1.8, to find out which the time and hour keys are, using the format 'yyyyMMdd' and 'HHmmss' from *DimDate* and *DimTime* respectively.

Next the stream continues in Figure 30 by focusing on the construction of multiple URL requests with embedded coordinates and with the help of APIs, retrieving the necessary information.

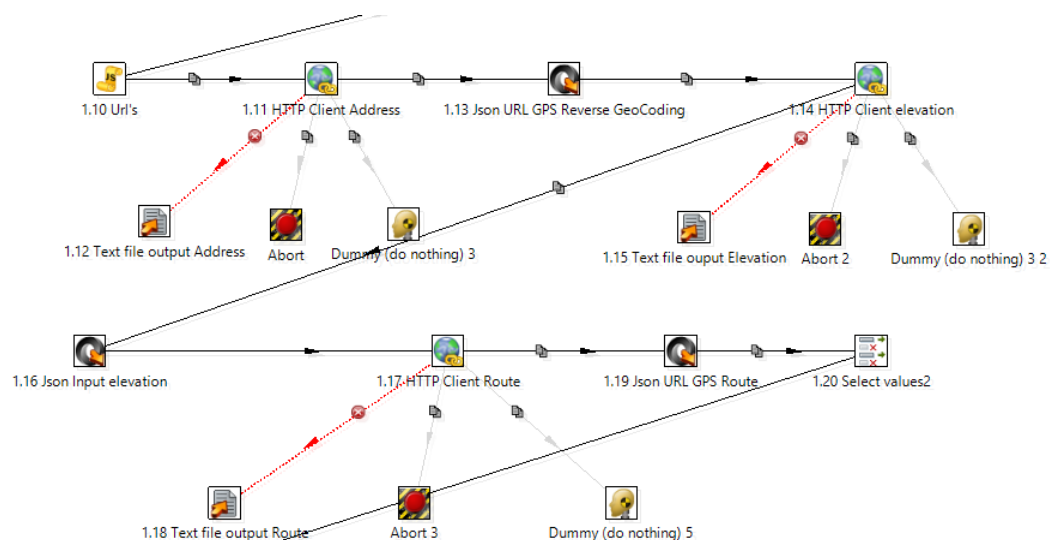


Figure 30 – APIs Request (Fact_GPSpart 2)

Therefore, an API can be considered as: *“An API defines an abstraction layer for software services and resources. The API describes the rules and the expected behavior so that the service or resource can be accessed as a black box, without a need to understand the details of how it is implemented.”* [Cheshire, 2011]. So, a web API, is an application programming interface (API) for either a web server or a web browser. The code can run on client-side, meaning that the builded application runs on the local machine instead of remote or other machine (Server-Side). Server-Side can also be considered as web service, or a SOA (Service-oriented architecture based application), or REST based applications [Oriani, 2014].

Step 1.10 from Figure 30 aims to encode URL's to be used with the web API. The use of this component can be considered as an extension of the component present in step 1.11, since it is necessary to use a dynamic address to process the different gps rows coordinates. So, following this approach, step 1.11 receives an URL that changes part of its content (gps coordinates) in each interaction in order to get the corresponding information about gps location using the API. Otherwise, it would not make the correct API request, since it would assume that the different rows of a table would all have the same address, which is ultimately not a true statement.

Since API retrieves information, whether in XML or JSON format, it was given preference to JSON because it is the current widely adopted standard [Aina, 2014]. Thereby, step 1.11 was implemented to extract targeted information (commonly known 'JSON parse') from response. For example, to get the name of the country from JSON response, it is `$.Response.View.Result[0].Location.Address.Country`.

Currently the *HereMaps* don't support REST web service of elevation, which is a fundamental parameter to fuel consumption. A proposed solution was to use the services of MapQuest. This particular service is free under the terms of MapQuest. To use it, it is necessary to do a registration and get a free AppKey. After that, depending on the availability of the service, it is only necessary to construct the requests. The current URL encoded allows for the developer to retrieve information about elevation given a latitude/longitude pair in JSON or XML formats. However, the coverage area is limited between 56° N and 60° N [MapQuestAPI, 2015].

The *Here Routing API* from Nokia maps provides routes information between location points, such as real-time traffic updates. Essentially, it offers the functionalities of calculating a route for a set of waypoints update previous calculated route and calculate an area which can be reached given time/distance travelling. Some of the capabilities useful to mention are speed categories, road condition, toll roads, direction of traffic flow (one-way, two-way) and blocked passages. As an addition, the truck attributes are also relevant, like physical restrictions (such as weight, height, length) or hazmat restrictions (trucks with hazardous materials forbidden or explosive and flammable). This service also provides customisable options given the type of vehicle. It ranges from public transportation (truck, car, pedestrian), route type (faster or shortest) or taking in consideration current status of traffic conditions (flow, incidents) [HereMaps, 2015a] .

To sum up, the same logic is applied to construct the request acknowledged by step 1.14 and step 1.17 (coded request with JavaScript component (step.1.10), use HTTP client (e.g. step 1.17) and parse the response with JSON (step.1.19)). Since a web service is used, the API response cannot always be available, and thus the errors are stored in text files (e.g. error from API reverse geocode (step 1.11) is stored in text file (step 1.12)).

Continued analysis of stream Fact_Gps, comes in Figure 31 that continues to describe the process of creating dimensions *DimGeography* and *DimRoad*.

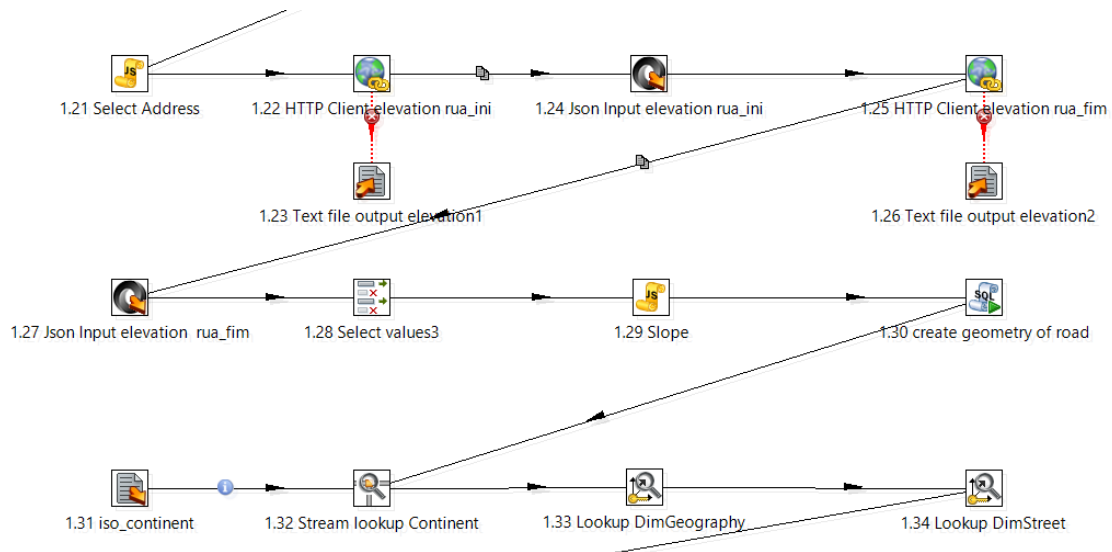


Figure 31 – DimGeography and DimRoad (Fact_GPSpart 3)

Since gps coordinates are not very accurate due to having EPE (Estimated Position Error) which varies from 5 to 15 meters (errors like receptor quality, factors caused by atmospheric layers, multipath (reflection/displacement) of the carrier signal wave due to natural or artificial obstacles, etc.). Since there is no algorithm implemented in the original system to correct it, an approach to select the fittest one was followed, given a radius from the centre of the gps coordinate, among three addresses.

From step 1.22 to step 1.28 information about road elevation is extracted, whereas it is retrieved the point where road begins and the point where it ends. The assumption is considered that each street between two points makes a segment and has a constant slop between the beginning and the end of the street (calculated in step 1.29). Step 1.30 implements a proposed solution to overcome the difficulty dealing with the geometry data type. Since the current software does not support it, an SQL query was applied. The geometry data type will enhance further specific search, like as giving a coordinate to understand which road segment it belongs to; or verify if a road segment is overlapped by another. These calculations of geometry data type are only possible with extension PostGis installed on system.

To complete the information, it was also necessary to merge the retrieved information from HereMapsAPI [HereMaps, 2015b] with continent details, since it was not provided by default.

Step 1.31 uses an external file containing continent and country information and matches it with the current stream (step 1.32). Wherefore, the dimension *DimGeography* gives information about country, city, and postal code among others. However, there is a need to perform queries at street level which lead to the implementation of dimension *DimRoad/DimStreet* which gives information about road name, recommended velocity, direction traffic, house numbers if exist, etc.[HereMaps, 2015b].

Then in Figure 32 the implementation of several components that lead to fuel consumption calculation appears.

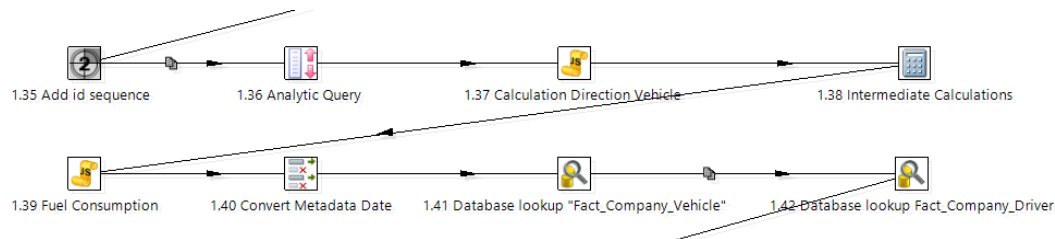


Figure 32 – Calculation Fuel Consumption (Fact_GPS part 4)

Table 24 – Description ETL process table Figure 32

Step	Operation Type	Description
1.35 Add id sequence	Get value from sequence	This step adds a field to the data stream. This sequence is used to group values in step 1.36
1.36 Analytic Query	Analytic Query	This step group values fields based on specific field value. It is used to compare values within previous rows allowing this way to make some mathematical operations.
1.37 Calculation Direction Vehicle	Modified Java Script Value	This step, apart from calculating the direction of vehicle between two coordinates, also does calculations to sign if a vehicle is stopped or in movement.
1.38 Intermediate Calculations	Calculator	This step is used to do some mathematical operations about velocity, time pass since last signal acquisition, difference between altitude, etc.
1.39 Fuel Consumption	Modified Java Script Value	This step uses parameters such acceleration, grade and constant value R_t (total reactive force) to calculate an approximate estimative about fuel consumption [Guo, Ma, Yang, Jensen, C.S., et al., 2012].
1.40 Convert Metadata Date	Select / Rename Values	This step convert <i>date_utcto</i> format "yyyy/MM/dd" and adjust latitude / longitude to have only 6 decimal digits.
1.41 Database lookup "Fact_Company_Vehicle"	Database Lookup Value	This step uses a temporary <i>factless</i> table (table only with foreign keys) to search the new current key of company and vehicle given original vehicle key.

1.42 Database lookup "Fact_Company_Driver"	Database Lookup Value	This step stores a driver who is assigned to a company, allowing to retrieve the new keys.
---	--------------------------	--

Step 1.35 is a complement of step 1.36 which focuses on aggregations. In this case is used to aggregate previous rows with the purpose of calculating the difference between points of elevation, time pass between acquisition since last gps coordinate, distance travel, etc. There are used instructions LAG and LEAD rows to accomplish these calculations.

Step 1.37 uses a JavaScript component where calculations of high, vehicle direction and indicative of state of vehicle are present. The vehicle direction is to give a point of reference if further calculations are needed related to the road. In the same component, it is also where conditions of velocity and ignition are checked. In addition, it also verifies if the field velocity from the system respects the recommended range velocity.

Step 1.39 uses vehicle speed v_t (m/s) and acceleration a_t (m/s²) and road grade θ_t (%) at time point t , and computes the result f_t (mL/s).

$$f_t = \begin{cases} 0.444 + 0.9 \cdot R_t \cdot v_t + [0.05 \cdot 4a^2 \cdot v_t]_{a_t > 0} & R_t > 0 \\ 0.444 & R_t \leq 0 \end{cases} \quad (1)$$

The $R_t = 0.333 + 0.00108 \cdot v_t^2 + 1.2 \cdot a_t + 0.1177 \cdot \theta_t$ is the total reactive force required to drive the vehicle [Guo, Ma, Yang, Jensen, C.S., et al., 2012].

The step 1.41 consults a temporary table where are registered all relations between company and vehicles. So, using the vehicle *id*, it is possible to identify the vehicle and company. The step 1.42 follows the same logic. Here, a lookup for driver key is made to find out which vehicle was associated given a specific day.

Finally, in Figure 33, some table inputs are displayed as well as the loading of the fact table.

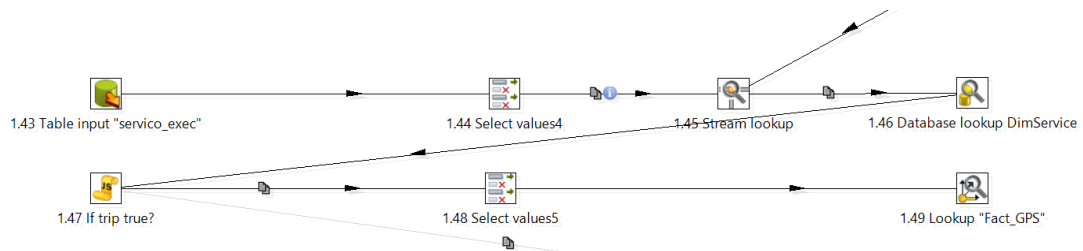


Figure 33 – Loading Fact table (Fact GPS part 5)

Step 1.43 extracts information about a service. A service is a planned trip assigned by a manager of orders. The component in step 1.45 uses the new key created within a system by looking up in service source database, and tries to make a correspondence. If no value was found, it means that it is an unplanned trip, and therefore should be treated as warning. This

is made by step 1.47 which identifies as planned/unplanned with *Boolean* field. Finally, the step 1.49, loads the data into the table “Fact_GPS”.

6.1.5 Loading Process

To load information into tables it is necessary to use specific components from PDI. There are many ways depending on the purpose to achieve, whenever inserting or updating data in the database. One requirement, as it was mentioned in each dimension, was to create a surrogate key/foreign key that could keep referential integrity between all database tables. In other words, the fact table cannot contain corrupt or unknown foreign key references. So, the method employed in this project was to create in each dimension an auto-number which increments when a new record is inserted, except in *DimDate* and *DimTime* that use semantic keys. In addition, it is normal to use numeric type since chars/varchar are less efficient when joins are made [Kimball and Ross, 2013].

Other important step before loading is the way how actually data is stored. In some cases, the possible existence of previously entered data may even happen, which can be a problem if duplicated data is inserted. Therefore, it is here that the mechanism of SCD (described on subchapter 3.6) stands, since it captures the changes made to data overtime [Kimball and Ross, 2013].

Remarkably, as the powerful tool that it is, the Pentaho Data Integration provides components according to practises of Kimball methodology. In previous figures presented in the transformation process, they share the component present in Figure 34 or Figure 35 in the end of the stream. Mainly, as the names states, it combines the search process with update, offering an easy solution for implementing the BI concepts.

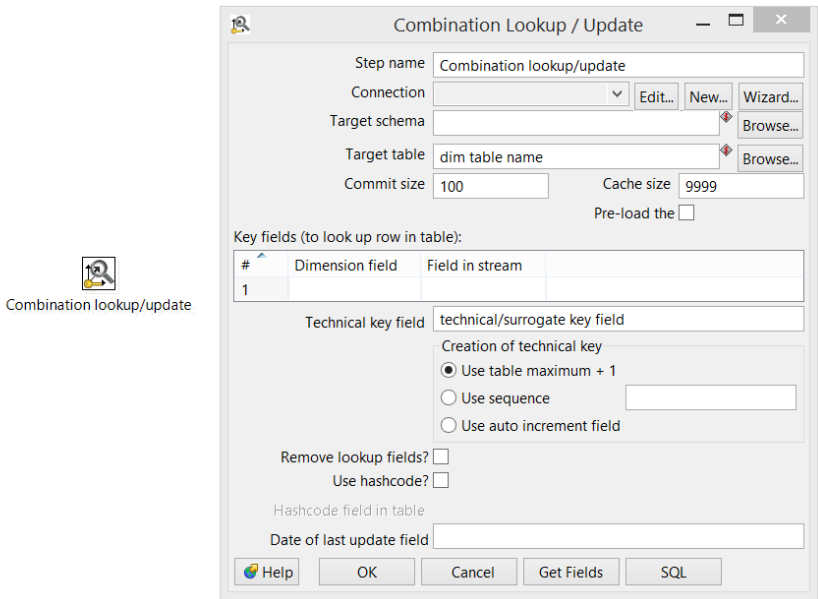


Figure 34 – Combination Lookup/Update: visual design (left), menu settings (right)

Generically, Figure 34 starts by looking up combinations of business keys from different rows present on the target table. Then, if this combination of business key fields already exists, return its surrogate key (identified in Figure 33 as technical key). If this combination of business key doesn't exist yet, a row is inserted with the new key fields and returned its (new) surrogate key. At the end, all input fields are putted on the output stream including the returned surrogate key [PentahoWiki, 2010].

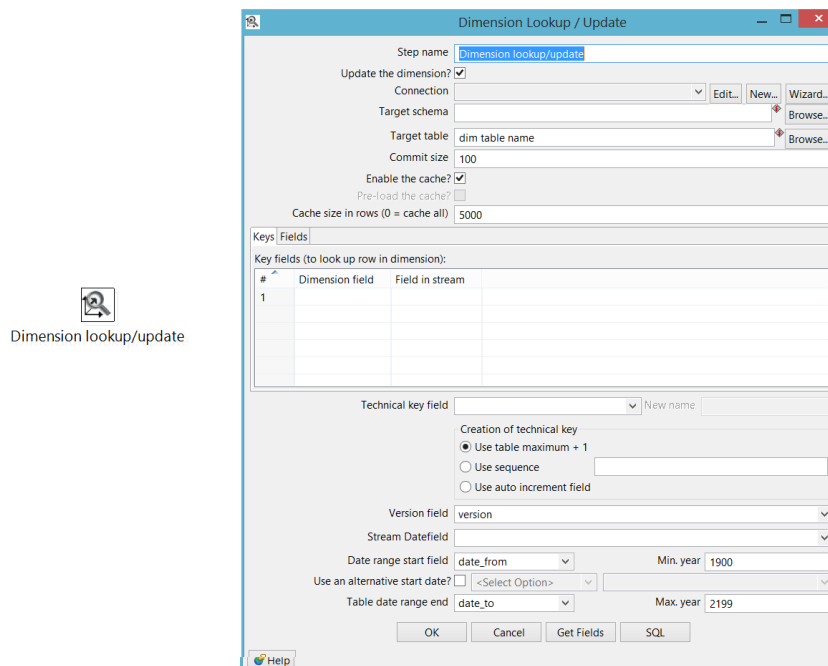


Figure 35 – Dimension Lookup/Update: visual design (left), menu settings (right)

Similar to Combination Lookup/Update, the Dimension Lookup/Update step allows implement Ralph Kimball's slowly changing dimension for both types: Type 1 (update) and Type 2 (insert) together with some additional functions [PentahoWiki, 2010]. Since this component was primarily used with Type II, it was indispensable to add three more columns in each corresponding dimension table. It uses date fields (start and end date) and version in order to keep track of changes, where each time a record is updated, the number of version is incremented as well, where the highest number corresponds to the current version of the record.

6.2 Main ETL Stream

A good practice in any ETL Architecture project, is to subdivide a big problem into smaller ones[Becker, 2009]. So, after dealing with ETL processes, a stream of Figure 36 was created, representing an abstraction layer of the full project. Here is visible a sequence of processes to be executed mainly composed by Jobs (orange arrows in Figure 36) and Transformations (green arrows in Figure 36) with specific purposes. Whereas Transformation focuses on moving and transforming rows from source to target, Jobs are more about high level flow

control: executing transformations, sending mails on failure, transferring files via FTP, etc. Another key aspect is in this stream where global variables are set, as well as the structure of tables and restrictions. Moreover, to improve performance is a good strategy implement indexes since they are as enhancers of query time [Oracle, 2015]. Therefore, Table 25 gives a brief description of each component implemented.

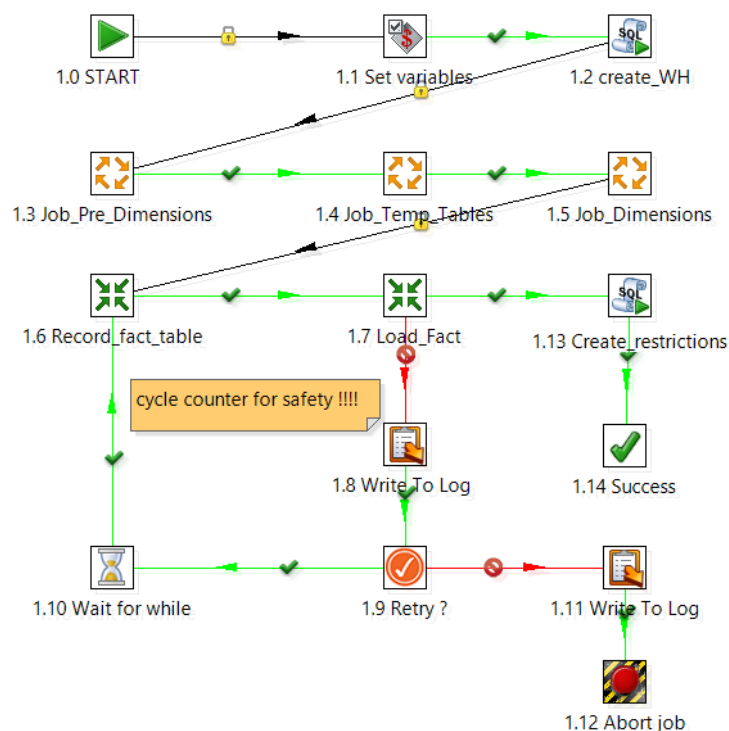


Figure 36 – ETL Main Job

Table 25 – Description of Main ETL process

Step	Operation Type	Description
1.0 START	Job Scheduling	Initiate the execution of program. It can be schedule to execute operations daily, weekly, monthly at specific time of day.
1.1 Set variables	Set Variables	This step assigns to a virtual machine the environment variables. Database name, password, username, and database host are defined here.
1.2 create_WH	Execute SQL Script	This step runs an external SQL script which allows to create tables in the data warehouse.
1.3 Job Pre Dimensions	Executing a job	This step execute the loading of dimensions time and date.
1.4 Job Temp Tables	Executing a job	Create a temporary database to improve performance of data loaded into data warehouse
1.5 Job Dimensions	Executing a job	Is a job which executes the loading of data into dimensions.
1.6 Record fact table	Job entity	This step verifies if the table “Fact GPS” has data.

	transformation	In case it is true, it stores the last record inserted into file.
1.7 Load Fact	Job entity transformation	Runs the transformation of Fact table.
1.8 Write to Log	Write to log	This step writes to log during execution in case of error occurrence.
1.9 Retry?	Simple Evaluation	This step counts the number of errors detected, usually related with APIs.
1.10 Wait for while	Wait for	Delays the execution of stream according to stipulated time
1.11 Write to log	Write to log	Write to log the number of attempts where the step 1.9 has fault condition.
1.12 Abort Job	Abort Job	Abort the current execution of job.
1.13 Create restrictions	Execute SQL Script	This step creates restrictions between tables in system (foreign keys).
1.14 Success	Success	Component that identifies the execution of successful stream.

The stream starts by creating the environment variables. This could be accomplished in many ways with more or less security depending on the purpose. With component step 1.1, they are locally assigned whereas a future improvement may read them from an external file. Then in step 1.2 an external SQL file is read, which proceeds to the creation of dimensions and fact table. Next, we have step 1.3 that executes a Job, where time and date dimensions are loaded into the data mart.

Step 1.4 is a Job that contains several transformations which create temporary tables from the original tables that have more data loaded. It divides data from source by month into these temporary tables with the goal of improving query performance.

Step 1.5 holds the several ETL processes previously mentioned on subchapter 6.1 – ETL Implementation. Using this design allows the developer to have a greater control and order how transformations are handled facilitating the detection of issues during each ETL stage. Moreover, it gives a better understanding about the division of tasks.

The step 1.6 is a mechanism developed to trace the changes made in the source system and reflect them on the data mart. Wherefore was needed an approach which stored last update made, in order to allow incremental updates. The component has the following logic: checks if there is no written information on the table; in case that is true, it inserts the data into a table and then writes the last record inserted into a file; otherwise, it is necessary to check the last record inserted and proceed with the new insertion in the table.

Since the ETL process in step 1.7 uses REST services, it is reasonable to assume that the web service will not be always available. So, it might raise problems such as service timeout, request not found, bad request, unauthorized, etc. Therefore, the current mechanism creates a cycle within step 1.6, 1.7, 1.8, 1.9 and 1.10, in order to avoid that the program gets stalled by applying a waiting of 3 seconds (step 1.10) and writing an event to log, and restarting the

data loading into the fact table. In the worst case scenario, if the execution catches many errors during the execution, then step Abort (step 1.12) is triggered and is required to re-execute the program manually. If all goes as planned, then the primary key – foreign key constraints relations between dimensions and fact table are created (step 1.11).

6.3 OLAP Implementation

After storing the data into Data Mart in Postgres data base, is used the development tool Pentaho Schema Workbench (described on page 31) to produce the OLAP schema. Since it uses a designer interface to create OLAP cube schemas; a configuration of connection with Postgres is required in order to make it work. To configure data base connections go to: Option -> Connection. Then an identical menu configuration is shown, present in Figure 19 of subchapter 6.1.1 – Configuration Connection, which is configured the same way as indicated.

6.3.1 Build Cube

The first step to create a cube under Schema is to name it, and then add the dimensions. It is normal to find multiple cubes interconnected with different dimensions in order to accomplish more sophisticated and specific queries (designated as galaxy data warehouse) [FolksTalk, 2010]. Since this project focus on exploring the tool, it has generated only one cube following the structure displayed in Figure 37.

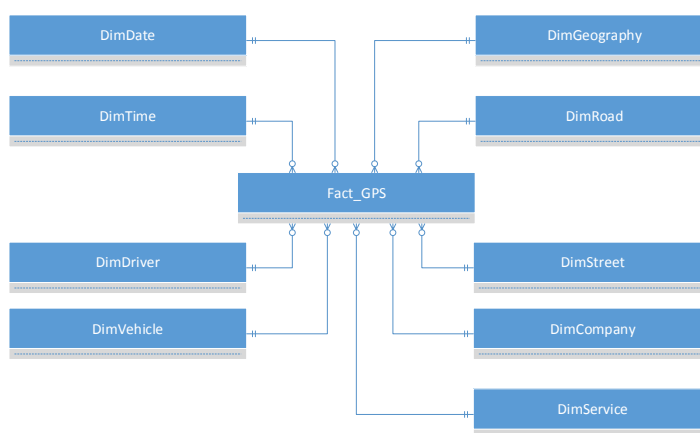


Figure 37 – Dimensional Model (simplified)

To start building a cube it is required to add a fact table (Fact GPS) followed by dimensions (in Figure 37 there are represented with the prefix “Dim”). For example, to add a dimension to the schema, press the button “Add dimension” showed in. The next step is to configure the desired hierarchy in each dimension (referenced on Pentaho Schema Workbench on chapter 4) according to relevance possible search. Then, the “Dimension Usage” allows adding and configuring the relations between fact table and dimensions with the help of a foreign key. For example, the Dimension Usage of Date is named as *DimDate* and the foreign key is the

date key. Thus, all dimensions present in Figure 37 are connected with the cube using this method, displayed in , which represents part of the structure of the cube, likewise an XML pseudo-code (Code 5) is generated.

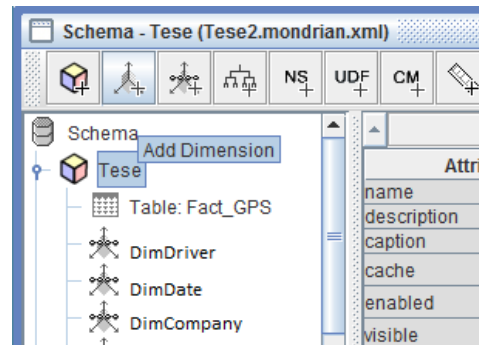


Figure 38 – The structure of Cube

```
<Schema name="Tese">
  <!-- Dimension Description; type: identify type dimension (geography,
  time or standard-->
  <Dimension name="DimDate" type="TimeDimension">
    <!-- add code to configure this dimension -->
    <Hierarchy name="All Date" hasAll="true" primaryKey="date_key">
      <Table name="DimDate" schema="public"/>
      <Level name="Ano" uniqueMembers="false" column="ano"
      levelType="TimeYears" type="Numeric">
        <!-- format date type -->
        <Annotations>
          <Annotation name="AnalyzerDateFormat">[yyyy]</Annotation>
        </Annotations>
      </Level>
    </Hierarchy>
  </Dimension>
  <!-- add other dimensions -->
  <Cube name="Tese">
    <Table name="Fact_GPS" schema="public"/>
    <!--add DimensionUsage for other dimensions -->
    <DimensionUsage name="DimDate"
    source="DimDate"foreignKey="date_utc_key"/>
    <!-- add other measures -->
    <Measure name="Km_end" column="km_acc"
    datatype="Interger"aggregator="max"/>
  </Cube>
</Schema>
```

Code 5 – XML Pseudo-Code Schema Mondrian

The Code 5 is part of the extracted XML code generated by the program Pentaho Schema Workbench, and has the peculiarity of being able to be edited in any text editor, as long as that tag structure is respected. Otherwise, the program generates a warning of incorrect content, forcing the developer to review each line.

When a vehicle performs a trip, it will store the Km value travelled in the database for several times during the course. By grouping the various Km readings under certain conditions, it is possible to reach a value. The presented measure illustrated in Code 6 uses a Calculated Member that aggregates rows.

```
<CalculatedMember name="Total Km"formatString="#.##"
    Formula="[Measures].[km_ini]-[Measures].[km_end]"
    Dimension="Measures"
    Visible="true">
</CalculatedMember>
```

Code 6 – Example Calculated Member

After finishing the construction of the OLAP cube, we proceed to the data analysis using MDX (Multi-Dimensional expression) queries. Therefore, the Mondrian schema applies a table mapping between MDX and SQL, whereas the OLAP engine will translate MDX into SQL and then query database server [PentahoWiki, 2010].

6.3.2 Deploy Mondrian Schema

Since Mondrian Schema works like a third-part software, it is required to commit the schema to Pentaho web server which contains an OLAP engine that runs Mondrian Schema. After upload it, is possible to explore the generated cube with an installed extension called Saiku by displaying data across multiple dimensions or even to build a dashboard.

To publish the schema it is necessary to follow the following steps:

1. Click “File” and then “Publish” to show the pop-up Publish Schema dialog, as it is shown on Figure 39.
2. Input Pentaho web server URL, user and the password. It is essential that the user has the necessary privileges to perform the operation.
3. The input JNDI Data source references which is the database connection of data mart. Then click “Publish” to finish.

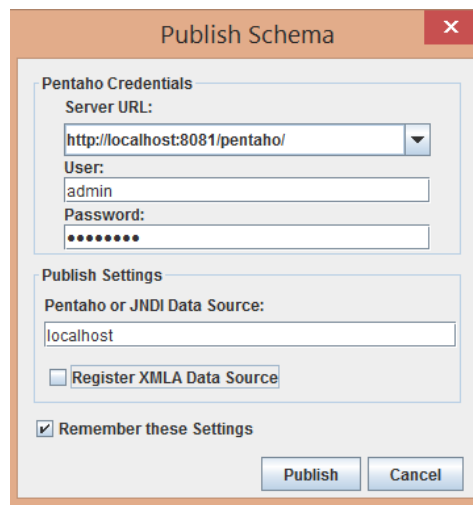


Figure 39 – Publish Schema Dialog

6.4 Pentaho Dashboard CDE

After the cube is published on the system, it is essential to use tools to visualize the data. This can be accomplished by any BI tool that supports the schema Mondrian (like TIBCOJasperSoft). Meanwhile, since Pentaho offers a roll of several tools for reports, visualization or data-mining, this section will focus attention only on the creation and operation of the construction of a Dashboard [Pentaho, 2015].

So, to build a dashboard having as its basis operation a tool Pentaho, can be achieved in many ways. One of the ways is using a paid version (Dashboard Designer), which offers an interactive visual analysis with drill through, filtering, zooming, etc. The carrying out of its construction, relies on drag and drop actions to achieve fast results of business user's key performance indicators; however, it restricts the solution to an already pre-designed structure. It is a very similar tool to the Saiku.

On the other hand, a more changeling way is to use the CTools (free version). It gives a wider range of options to customize dashboards since it uses Html, CSS, AJAX, JavaScript or even JQuery code [Webdetails, 2015]. Besides the facility to adapt the visualization to any display device (e.g. mobile, computer, PDA, etc.) it is an asset. Nevertheless, it requires careful and time-consuming learning curves to understand the tool's potential. Likewise, like in most of free software versions, the reliable documentation must be paid for, which, at the beginning, arises difficulties in the learning process.

Figure 40 displays the structure of the layout from a web page Dashboard, where the code can be all written in Html or use an assistant to create rows and columns divisions. The last method was chosen due to documentation encountered about CDE [Pochampalli, 2015], even though the software allows to use CSS and/or bootstrap classes.

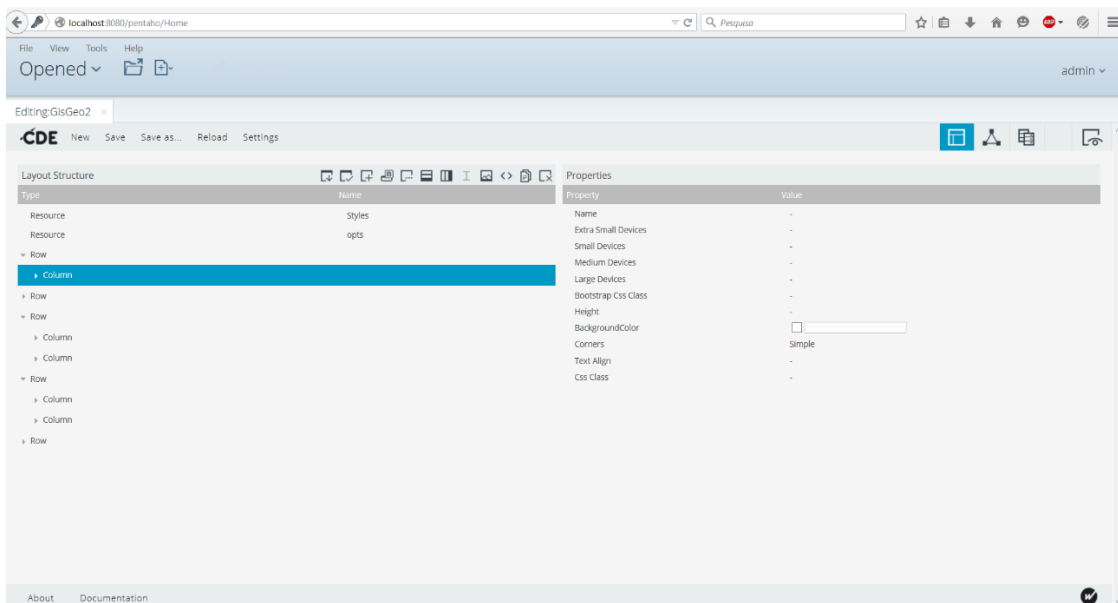


Figure 40 – CDE Layout Structure

After the layout is defined, there comes the choice of viewing components that display the information. The CDE offers a wide range of options like different types of charts, selects (radio button, multi-selection button, date input, etc.), popups, use custom scripts (functions) or even integrate a user-defined component, which is present in Figure 41. Furthermore, it is common to add the tool third-party components in order to give a more professional visual appearance to the created dashboard

To use a component from the list in Figure 41 (left side), it is only necessary to double click on it. Since each one belongs to a category, the CDE aggregates them in the centre, intending to make it easily accessible and identifiable. On the right side of Figure 41, the properties of the selected component are shown, since each one has its own general and advanced properties. Generically, to put a component working, a reference of the data source and the place of layout where it should appear is required. In contrast, the advanced properties extended the component abilities; make it interactive and customizable by allowing, for instance, the implementation of a Java Script code. This type of customization empowers to drill through the data and make it navigable between components. These operations require configuration of input parameters and execution of functions in pre/post-action clickable.

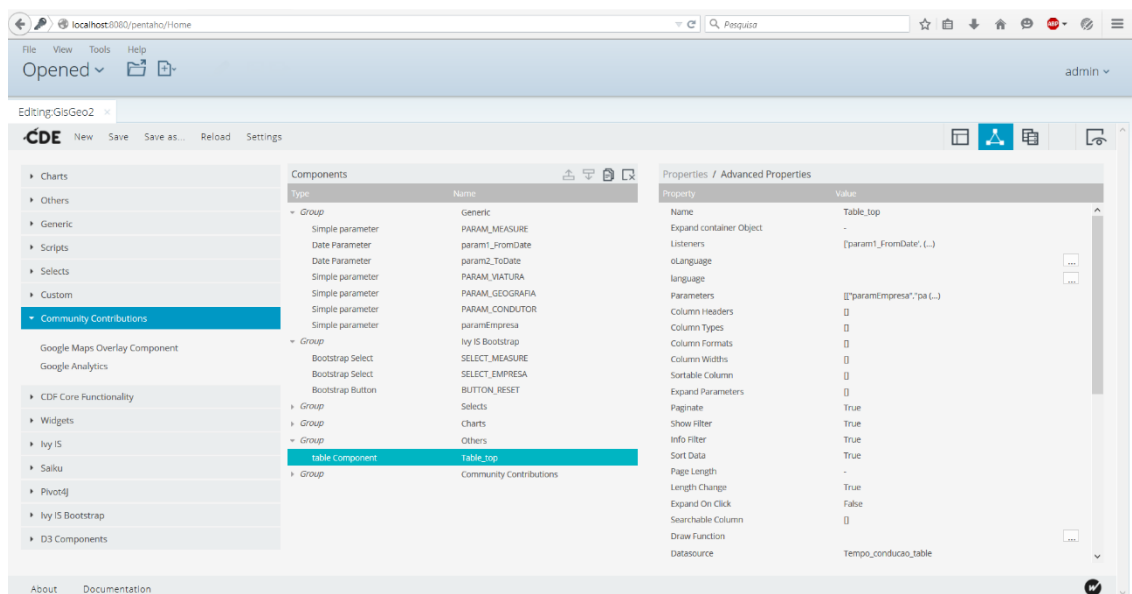


Figure 41 – CDE Components

In addition, Figure 42 demonstrates how connections are created. It supports different sources like MDX, SQL, MQL or KETTLE queries, which can be found in the menu on the left side. For this project, only SQL and MDX queries were used, in order to illustrate an example. In the first case, SQL query, it requires the Driver name, user, password and URL to access the database and retrieve data. There is an option in case of non-static input fields (configurable in “Parameters”) which makes a query modifiable. When using MDX query, JDNI (Java Naming and Directory Interface), Mondrian Schema and MDX code are necessary.

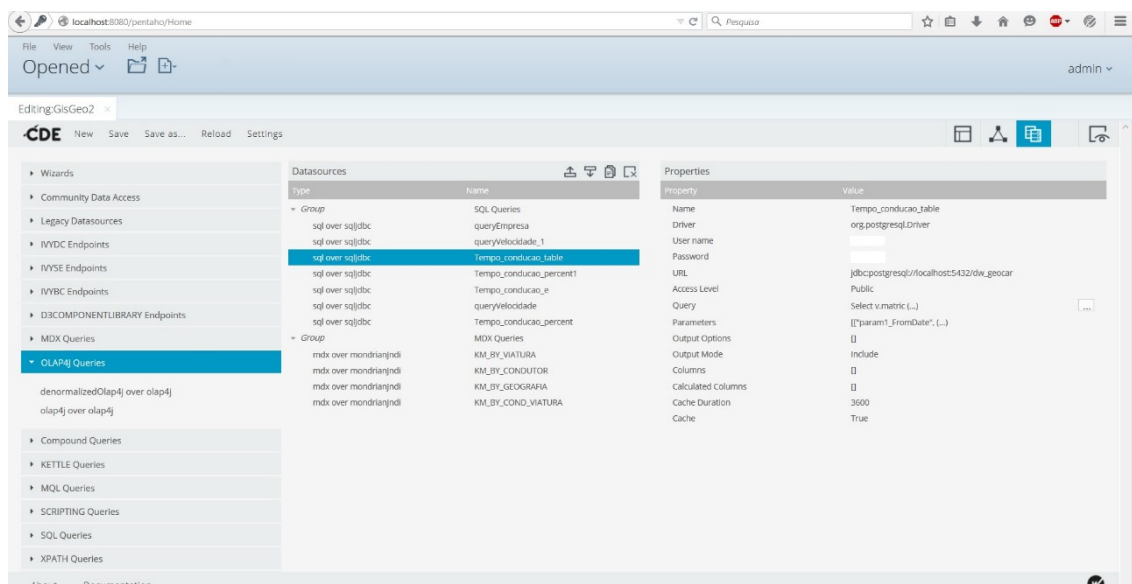


Figure 42 – CDE Data sources

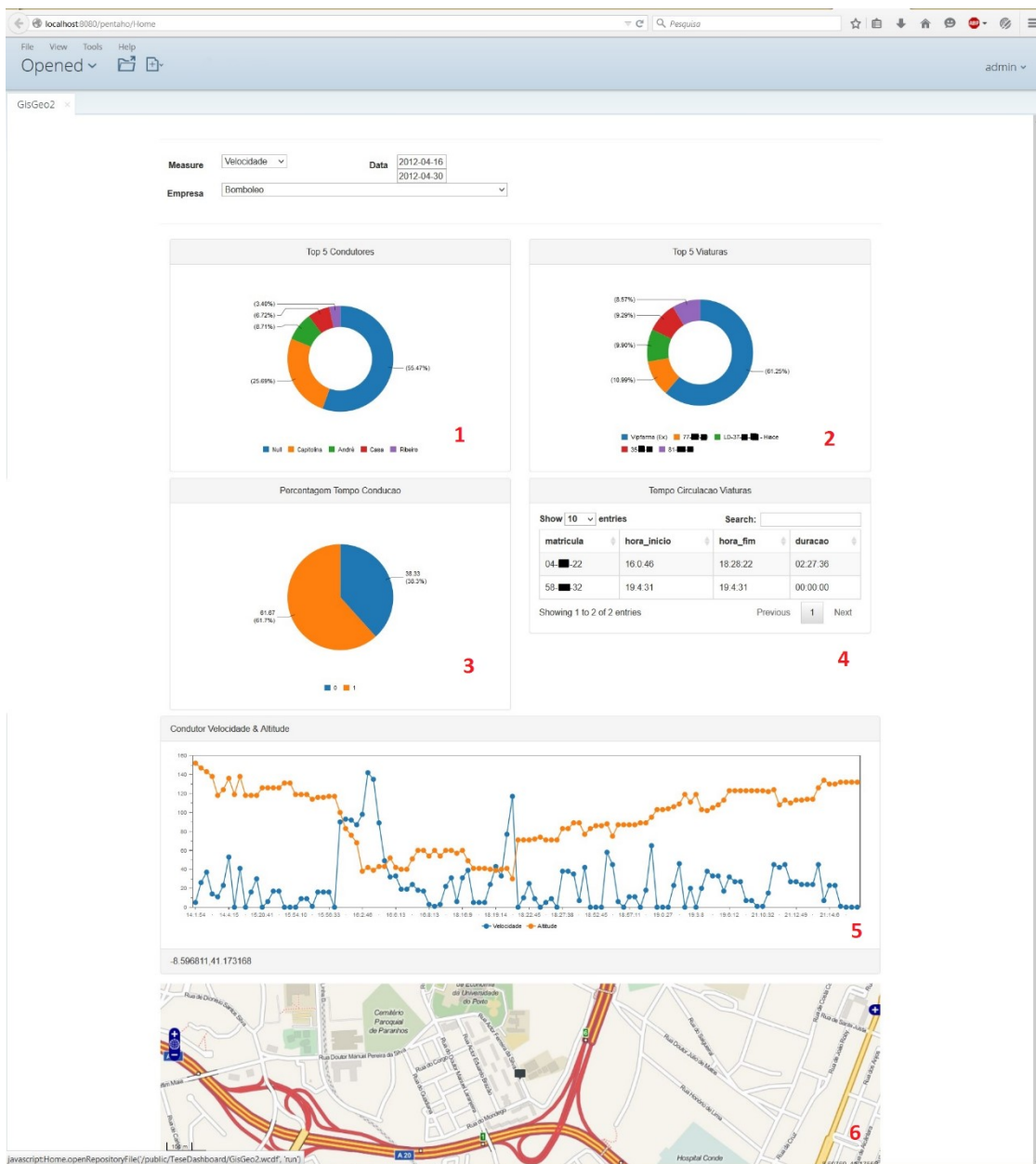


Figure 43 – Dashboard

Figure 43 displays the final result of an interactive Dashboard created where the main focus was to explore some chart abilities with selected KPI, and not a visual appearance. The developed dashboard is one of the ways to assist the fleet manager, by allowing to explore a specific pre-defined measure (Velocity, Fuel Consumption or Distance Travelled) and selecting a date. The date parameter popes a mini-calendar where a user selects the desired start date and end date to explore the data. With the aim to demonstrate the usefulness of the select button, there is also the possibility of selecting the desired company; but in a real case a specific manager of a company does not have access to this feature.

Then we have the first KPI which shows the *top five drivers* according to the selected parameter “Measure”. In this case, the pie chart rates the drivers who are leading to fast, which means, for example, there is the possibility these drivers have an aggressive driving. Therefore, in an analysis perspective, the best drivers are those that are not presented in the graph. The second chart displays the *top five vehicles* which had the higher speed. Again, it is possible to predict some conclusions about premature wear and malfunctions. Combining this information with a detailed analysis of historical records of vehicles it is possible to withdraw conclusions if the vehicle is still viable to be preserved in the fleet.

The graph number 3 from Figure 43 is a result of choosing a vehicle of graph number 2. In other words, the user selects a desired vehicle from the *top 5 vehicles* to consult information about percent of driving time. It is a useful measure because it gives the ratio between driving and stops, allowing to easily detect longer stops than usual expected. In graph number 4 a table is presented which displays all vehicles which belong to the company, each one represented by the registration. As well as being simple to detect vehicles that are moving on the specified data, it is also possible to see which one had more use. It can be an extra parameter that facilitates future assignment vehicle planning, balancing the utilization of the fleet.

Again, the result of selecting chart number 2, displays the result of line chart number 5 which represents the result of the selected vehicle speed during their daily activity. The blue line represents instant speed over time, and orange line the elevation. So, it is possible to study the behavior of the car/driving style. The dots on the graph lines are a matter of design, since the chart is clickable in order to select the value of time.

The final graph, a map, gets the selected value from chart number 5 and using a SQL query retrieves the vehicle's position from the database and draws it on the map. This approach allows recreating the vehicle's route with visual significance. Currently, the ability to recreate a vehicle's route may help in understanding the most possible viable paths to follow or in case of accidents.

6.5 Response to the Requirements

Next, it is presented a possible interpretation from the proposed requirements illustrated in Table 4 on page 43, wherein each is a transcription from the table and followed by the data analysis. The current data is a sample of the full database, in which some of the data have been altered for reasons of privacy. The results obtained were generated via pivot tables with the aid of Excel 2013 software.

- Identify the company's activity by geographic area

The proposed solution is to build a graph that allows to select the desired company by region, given a specific date. A graphical solution can be found in Figure 44.

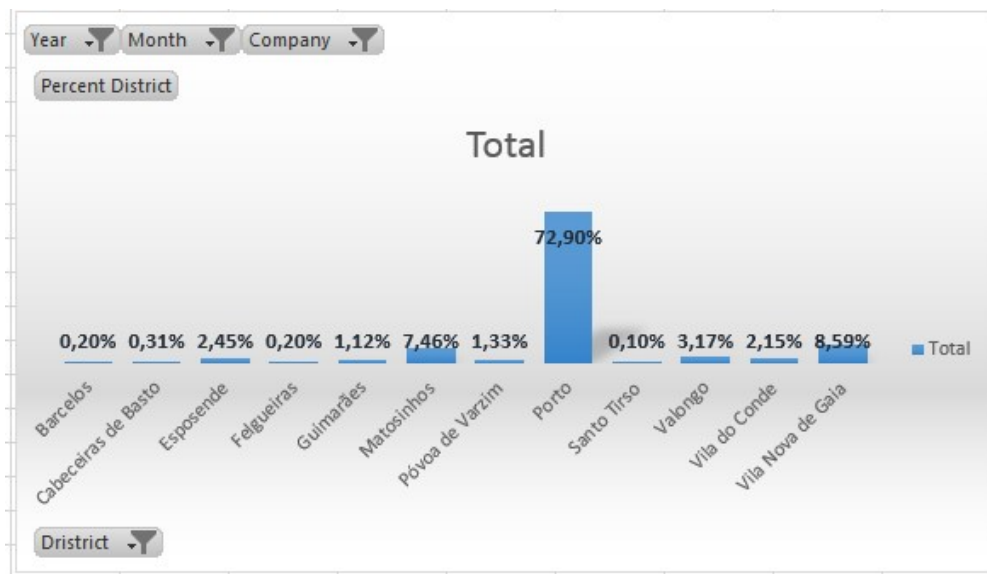


Figure 44 – Response to Requirement 1

According to Figure 44 is possible to consult company activity by district. This metric is important because gives an overall understanding what district of specific company held more business.

- Identify which roads that are frequently used by kms and/or time and/or type of vehicles

The second requirement, requires aggregation and count of several rows and rank them either by length, time of travel in the road segment or count vehicles that have pass through. Figure 45 is an example using a measure Km, that displays the most circulated road overall.

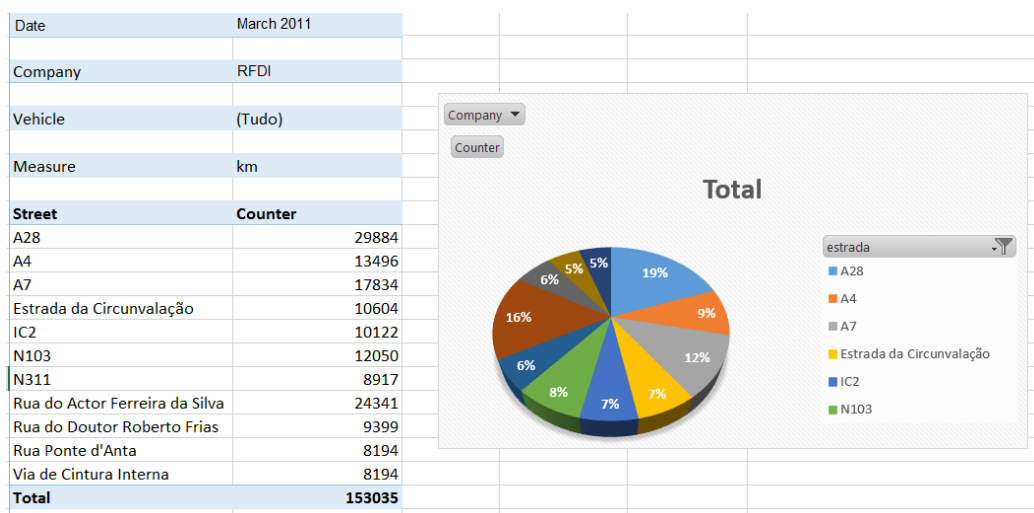


Figure 45 – Response to Requirement 2

From Figure 45 is displayed an example of the Top 10 roads most circulated from company RFDI on March 2011. The cumulative result is the total number of kilometres travelled by vehicles of this company. The same results are also represented in the graph, in order to make them more evident.

- Know if the driver drives mostly at night or day; how it influences on fuel consumption and distance travelled

The third requirement explores driver situation about fuel consumption. Since vehicles behave differently, is complicated to get accurate results to compare and since a driver behaviour is also an important factor, it is important to keep in mind these criteria's when are making a critical analysis of the results. Thereby, Figure 46 shows Top three drivers, indicating the percentage associated to dawn and to afternoon.

Company	RFID			
Fuel Consumption	Driver			
Hour	Carlos	João	Paulo	Total
AM	6,15%	6,15%	5,59%	17,90%
PM	28,22%	28,22%	25,66%	82,10%
Total	34,38%	34,38%	31,25%	100,00%

Figure 46 – Response Requirement 3

- Vehicles go further without supplying fuel

The fifth requirement was not fully implemented since the table where the supplies of each vehicle are stored, is not implemented in the current system. Therefore, reworking the part of the requirement is possible to indicate which vehicles have more fuel consumption, and compare them with the accumulated Km. Then, in Figure 47, is shown the result of selecting 10 vehicles from a company in the month of March 2011.

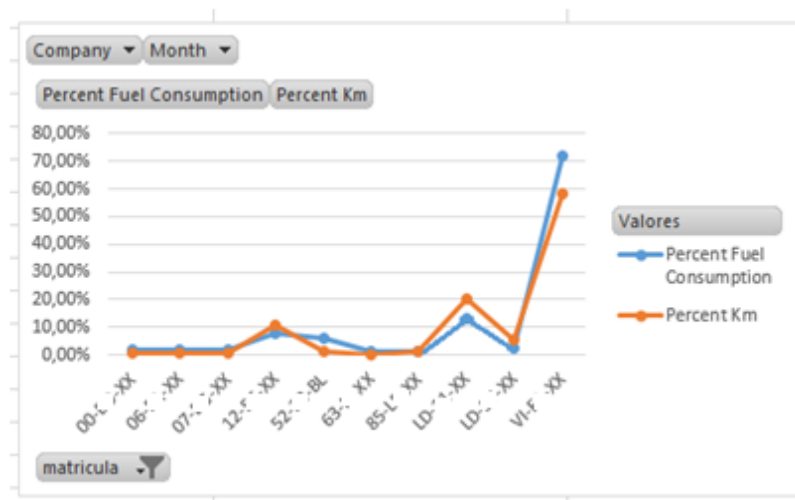


Figure 47 – Response to Requirement 5

The results presented in Figure 47, seeks a relationship between the distance traveled and fuel consumption value. Therefore, observing the graph, it appears that some of the vehicles have a higher consumption relative to distance traveled. It will be necessary to examine in more detail the vehicles in question, to understand what causes these differences. For example, may be associated with mountain journeys made or even the driver's own driving style.

- Identify the drivers that use vehicles for personal use

The sixth requirement is also essential because usually tries to avoid situations like: drivers are using them for side jobs or weekend getaways; or use the vehicle to do a much longer trip then previously scheduled. Thus, this metric is also important for this type of business model, identified by example in Figure 48.

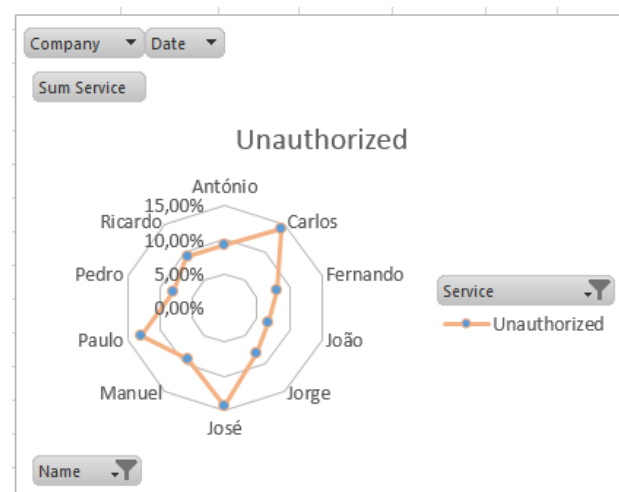


Figure 48 – Response to Requirement 6

Figure 48 displays the percentage of drivers who drove illicitly. For example, of all trips that the driver Carlos held, 15% of them did not agree with the planning. This margin may be associated with route deviations, or own use.

6.5.1 Software Versions

Table 26 shows the versions of the software used to develop this project as well as in Table 27 the experiment condition.

Table 26 – Software Version

Software	Version
Microsoft Excel	2013
PostgreSQL	9.1
PostGIS	1.5
Pentaho Data Integration	5.3
Pentaho Schema Workbench	3.9
Pentaho Business Analytics	5.3
Saiku (extension Pentaho BI)	2.0
CTools	5.3

Table 27 – Experiment Condition

Category	Item
Development Tools	Pentaho BI suite
Database	PostgreSQL+PostGIS
Server	Localhost
Computer Language	JSON, SQL, JavaScript

7 Conclusion

The idea of tracking vehicles is not new in the market, and the solutions offered by competitors are quite competitive. For this small growing company, have a solution that encompasses less costs to develop and can, at the same time, compete with others, is an asset. So, due to his dimension and policy, the company seeks innovative solutions.

In every development project, even a small BI project, is not an easy task to be accomplished. In each stage of modelling and developing a solution, it goes through many phases to become robust and trustworthy. Even though, is not assured of a successful implementation. So, understanding the concepts adjacent to BI solution is a starting point for creating better planning. This should be done accurately because it is a crucial piece to the success of such projects, and not take it lightly.

Another step that contributes to the success of the solution is the identification of requirements. At beginning they may be defined poorly and generically. It can be overwhelming to choose and identify them correctly, since there was a lack of experience with GIS (geographic information system).

From the literature founded, was possible to understand the dimension of the problem and give the necessary boost to clarify Ideas. However, very few literature offered a proper solution to this problem, since the Business Intelligence combined with GIS is relatively a new topic.

The proposed BI solutions from literature, were often used open-source tools. Therefore, when choosing the tool to develop the project, this was a strong argument. It was done a small study to verify the popularity among users, beside the features, since these are that contribute to the improvements of the tool, making it more desirable to use.

During the implementation phase, some difficulties are risen. First, was necessary to get familiar with the tool, since to date, was unknown. After overcoming this obstacle, was

followed the Kimball practices to develop a robust structure. After the ETL stage, the construction a data mart and creation of the OLAP cube, another complication appeared. How to display data stored in database? The first approach was to create a simple iterative dashboard, to allow rapid identification of some generic KPIs. This helped to gain insight into how a dashboard works and to understand the difficulties to build it. The second, was to use Excel to respond to the proposed requirements.

Overall, the BI concept also can be applied to GIS empowering better decisions in business, as it was proved with his dissertation. Thus, each company can set its indicators, whether they are profit maximization, process optimization or any other, aligned to the company strategies. Thereby, the proposed requirements, obtained a positive response, as proof of useful BI application.

In conclusion, with this work is possible to extend the literature found, serving as a basis for more complex developments in GIS. Plus, this helps to understand why many companies search for similar solutions to implement on their own businesses.

7.1 Limitations

Following are some limitations that were encountered through the project. Some of them were implemented because time constraints, and others were not considered, but they will bring an added value to the project.

The *DimTime* table can be extended with identifications of half hours only, of each half hour for the entire day or quarter hour for the entire day. In *DimVehicle* is used a default value 0000 to assigned a vehicle which have no year model listed on the source table. The right approach should be to store it into a log file, and later on, the value be manually corrected. The stream of fact table, currently only accepts Portuguese geocodes, and has that specific format. For example, if the postal code is from Spain, implies that every line associated is discarded.

7.2 Future Recommendations

Besides correcting the previous enumerated limitations, it should be inspected the main ETL stage, for example, to study the advantage of separating data into tables by month instead of having all the historical records into one table (better query performance vs space required). Another proposed solution is to use a third-party tool (HERMES²) to process GPS points before being uploaded to the fact table or even create grids, in order to extend geographic ability's. Finally, is necessary to find an alternative, at the enterprise level, to acquire altitude

² Available at: <https://hermes-mod.java.net/>

information, since the actual request ratio is low, or see if On-Board Diagnostic equipment's has that feature implemented, but not activated.

References

- 1KeyData (2015) *Data Warehousing Concepts*. [Internet] Available from <http://www.1keydata.com/datawarehousing/concepts.html> [Accessed 7th August 2015].
- Aina, E. (2014) *Why JSON Is Better Than XML*. [Internet] Available from <http://blog.cloud-elements.com/json-better-xml> [Accessed 7th September 2015].
- Andersen, O., Krogh, B.B., Thomsen, C. and Torp, K. (2014) An Advanced Data Warehouse for Integrating Large Sets of GPS Data. *Proceedings of the 17th International Workshop on Data Warehousing and OLAP - DOLAP '14* [Internet], pp.13–22. Available from <http://dl.acm.org/citation.cfm?doid=2666158.2666172>.
- Anon (1983) *Richard Stallman's personal site*. [Internet] Available from <https://stallman.org/biographies.html> [Accessed 7th August 2015].
- Barber, T. (2015) *Saiku Project Information*. [Internet] Available from <http://wiki.meteorite.bi/display/SAIK/Saiku> [Accessed 10th August 2015].
- Becker, B. (2009) *Six Key Decisions for ETL Architectures*. [Internet] Available from <http://www.kimballgroup.com/2009/10/six-key-decisions-for-etl-architectures/> [Accessed 15th September 2015].
- BetterBuys (2015) *Better Buys | Objective Software Reviews & Insights*. [Internet] Available from <https://www.betterbuys.com/> [Accessed 14th September 2015].
- Birt (2015) *BIRT*. [Internet] Available from <http://www.eclipse.org/birt/> [Accessed 7th August 2015].
- Bolduc, T. (2014) *What Is The OBD-II Port And What Is It Used For?* [Internet] Available from <http://www.makeuseof.com/tag/obd-ii-port-used/> [Accessed 20th August 2015].
- Boylan, C. (2015) *Business Intelligence*. [Internet] Available from <http://www.informationbuilders.com/business-intelligence> [Accessed 7th August 2015].
- Buttry, S. (2011) *The 5 W's (and How) are even more important to business than to journalism*. [Internet] Available from <https://stevebuttry.wordpress.com/2011/04/27/the-5-w%E2%80%99s-and-how-are-even-more-important-to-business-than-to-journalism/>.
- Buys, B. (2015) *The Definitive Guide to Business Intelligence*. [Internet] Available from <https://www.betterbuys.com/bi/definitive-guide-bi/> [Accessed 6th August 2015].
- Cheshire, M. (2011) *What is an API?* [Internet] Available from <http://www.quora.com/What-is-an-API> [Accessed 3rd May 2015].
- Columbus, L. (2015) *Key Take-Aways From Gartner's 2015 Magic Quadrant For Business Intelligence And Analytics Platforms*. [Internet] Available from <http://www.forbes.com/sites/louisacolumbus/2015/02/25/key-take-aways-from-gartners->

2015-magic-quadrant-for-business-intelligence-and-analytics-platforms/.

Developers, G. (2015) *Google Maps Web Service APIs*. [Internet] Available from <https://developers.google.com/maps/web-services/> [Accessed 20th August 2015].

Devens, R.M. (1864) Devens. In: *Cyclopaedia of Commercial and Business Anecdotes; Comprising Interesting Reminiscences and Facts, Remarkable Traits and Humors of Merchants, Traders, Bankers Etc. in All Ages and Countries*. [Internet] D. Appleton and company, p.210. Available from <https://archive.org/details/cyclopdiaofcom02deverich>.

Diffen (2013) *Snowflake Schema vs. Star Schema*. [Internet] Available from http://www.diffen.com/difference/Snowflake_Schema_vs_Star_Schema [Accessed 14th September 2015].

Edmunds (2015) *Edmunds Developer Network - Welcome to the Edmunds API*. [Internet] Available from <http://developer.edmunds.com/index.html> [Accessed 20th August 2015].

Energy, U.. D. of (2015) *FuelEconomy.gov Web Services*. [Internet] Available from <http://www.fueleconomy.gov/feg/ws/index.shtml> [Accessed 20th August 2015].

FolksTalk (2010) *Data Warehouse Dimensional Modelling (Types of Schemas)*. [Internet] Available from <http://www.folkstalk.com/2010/01/data-warehouse-dimensional-modelling.html> [Accessed 15th September 2015].

G2crowd (2015) *Compare Jaspersoft, Pentaho, Actuate BIRT*. [Internet] Available from https://www.g2crowd.com/compare/jaspersoft-vs-pentaho-vs-actuate-birt?starred_ids= [Accessed 20th September 2015].

Gabelica, H. (2013) *Agilna poslovna inteligencija*. [Internet] Available from <https://sqlbicro.wordpress.com/2013/02/13/agilna-poslovna-inteligencija/> [Accessed 10th August 2015].

Gartner (2013) *Business Intelligence (BI) Platforms*. [Internet] Available from <http://www.gartner.com/it-glossary/bi-platforms> [Accessed 15th September 2015].

GeoPostcodes (2015) *Postal & address database*. [Internet] Available from <http://www.geopostcodes.com> [Accessed 6th September 2015].

GisGeo (2013) *GisGeo*. [Internet] Available from <http://www.gisgeo.pt/> [Accessed 6th August 2015].

Guo, C., Ma, Y., Yang, B., Jensen, C. and Kaul, M. (2012) Ecomark: evaluating models of vehicular environmental impact. *Proceedings of the 20th ...* [Internet], pp.269–278. Available from <http://dl.acm.org/citation.cfm?id=2424356> [Accessed 26th November 2014].

Guo, C., Ma, Y., Yang, B., Jensen, C.S. and Kaul, M. (2012) EcoMark: Evaluating Models of Vehicular Environmental Impact. In: *Proceedings of the 20th International Conference on Advances in Geographic Information Systems, SIGSPATIAL '12*. [Internet] New York, NY, USA, ACM, pp.269–278. Available from <http://doi.acm.org/10.1145/2424321.2424356>.

- Hatfield, G. (2014) *10 Ways to Reduce Fleet Costs*. [Internet] Available from <http://www.fleetfinancials.com/article/story/2014/02/10-ways-to-reduce-fleet-costs/page/1.aspx> [Accessed 18th August 2015].
- Heinze, J. (2014) *A History of Business Intelligence*. [Internet] Available from <https://www.betterbuys.com/bi/history-of-business-intelligence/> [Accessed 6th August 2015].
- Heinze, J. (2015) *Compare Business Intelligence Software*. [Internet] Available from <https://www.betterbuys.com/bi/reviews/> [Accessed 15th September 2015].
- HereMaps (2015a) *Consumer Mapping*. [Internet] Available from <https://developer.here.com/solutions> [Accessed 6th September 2015].
- HereMaps (2015b) *Here Rest APIs*. [Internet] Available from <https://developer.here.com/rest-apis> [Accessed 18th August 2015].
- Hermannsson, J. (2005) *Real-Time Queries and Analysis on Moving Cars*. [Internet] AALBORG University. Available from [http://projekter.aau.dk/projekter/en/studentthesis/realtime-queries-and-analysis-on-moving-cars\(95fa0fb0-a790-45ee-bfd8-47e0989e6c61\).html](http://projekter.aau.dk/projekter/en/studentthesis/realtime-queries-and-analysis-on-moving-cars(95fa0fb0-a790-45ee-bfd8-47e0989e6c61).html).
- Holoch, R. (2014) *How to Use Pentaho PDI to Load (ETL) Data In and Out of Splice Machine*. [Internet] Available from <https://splicemachine.zendesk.com/hc/en-us/articles/203354587-How-to-Use-Pentaho-PDI-to-Load-ETL-Data-In-and-Out-of-Splice-Machine> [Accessed 10th August 2015].
- Hope, C. (2015) *When was the first computer invented?* [Internet] Available from <http://www.computerhope.com/issues/ch000984.htm> [Accessed 6th August 2015].
- Immanuel (2014) *33 Open Source and Free Business Intelligence Solutions*. [Internet] Available from <http://www.predictiveanalyticstoday.com/open-source-free-business-intelligence-solutions/> [Accessed 7th August 2015].
- Inmon, W.W. (1992) *Building the Data Warehouse*. First ed. [Internet] New York, NY, USA, John Wiley & Sons, Inc. Available from <http://www.amazon.com/Building-Data-Warehouse-W-Inmon/dp/0764599445/>.
- Jakobsen, K., Mouritsen, S.C.H. and Torp, K. (2013) Evaluating eco-driving advice using GPS/CANBus data. *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems - SIGSPATIAL'13* [Internet], pp.44–53. Available from <http://dl.acm.org/citation.cfm?doid=2525314.2525358>.
- Janssen, C. (2015) *Definition - What does Business Intelligence (BI) mean?* [Internet] Available from <http://www.techopedia.com/definition/345/business-intelligence-bi> [Accessed 7th August 2015].
- Jaspersoft (2015) *TIBCO Jaspersoft*. [Internet] Available from <http://www.jaspersoft.com/editions> [Accessed 7th August 2015].
- Jensen, A.F. and Larsen, T.V. (2010) *Travel-Time Estimation in Road Networks Using GPS Data*.

[Internet] Available from <http://projekter.aau.dk/projekter/files/61070977/1181652577.pdf>.

KeeResources (2015) *Free vehicle data for software and website developers*. [Internet] Available from <https://www.keeresources.com/data/free-vehicle-data/> [Accessed 20th August 2015].

Kent, W. (1982) *A Simple Guide to Five Normal Forms in Relational Database Theory*. [Internet] Available from <http://www.bkent.net/Doc/simple5.htm> [Accessed 14th September 2015].

Kimball, R. (1996) *The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses*. New York, NY, USA, John Wiley & Sons, Inc.

Kimball, R. and Ross, M. (2013) *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling*. 3rd ed. Wiley Publishing.

KimballGroup (2015) *Dimensional Modeling Techniques*. [Internet] Available from <http://www.kimballgroup.com/data-warehouse-business-intelligence-resources/kimball-techniques/dimensional-modeling-techniques/> [Accessed 4th September 2015].

Lachlan, J. (2014) *Defining Business Intelligence 3.0*. [Internet] Available from <http://www.yellowfinbi.com/YFCommunityNews-Defining-Business-Intelligence-3-0-159445> [Accessed 6th August 2015].

Loong, D. (2014) *How idling is calculated in the Trip History Page*. [Internet] Available from <https://helpdesk.geotab.com/entries/33992644-How-idling-is-calculated-in-the-Trip-History-Page> [Accessed 16th August 2015].

Luhn, H.P. (1958) A Business Intelligence System. *IBM Journal of Research and Development*, 2 (4), pp.314–319.

Mann, S. (2009) *Advanced Dimension Data Security with SQL Server 2008*. [Internet] Available from http://www.beyeblogs.com/rda_corp/archive/2009/01/advanced_dimens.php [Accessed 15th August 2015].

MapQuest (2015) *MapQuest+Developer*. [Internet] Available from <https://developer.mapquest.com/products> [Accessed 18th August 2015].

MapQuestAPI (2015) *Open Elevation Service Developer's Guide*. [Internet] Available from <http://open.mapquestapi.com/elevation/> [Accessed 28th March 2015].

Microsoft (2008) *Employee Table (AdventureWorks)*. [Internet] Available from <https://technet.microsoft.com/en-us/library/ms124432%28v=sql.100%29.aspx> [Accessed 25th August 2015].

Microsoft (2015) *MDX query basic*. [Internet] Available from <https://technet.microsoft.com/en-us/library/aa216770%28v=sql.80%29.aspx> [Accessed 7th August 2015].

Mitra, A. (2015) *The 101 Guide to Dimensional Data Modeling*. [Internet] Available from

<http://dwbi.org/data-modelling/dimensional-model/1-dimensional-modeling-guide> [Accessed 18th September 2015].

Mulcahy, R. (2007) *Business Intelligence Definition and Solutions*. [Internet] Available from <http://www.cio.com/article/2439504/business-intelligence/business-intelligence-definition-and-solutions.html#1> [Accessed 7th August 2015].

MySQL (2015) *MySQL*. [Internet] Available from <https://www.mysql.com/> [Accessed 18th September 2015].

Nunes, C. (2010) *Solução de Business Intelligence utilizando tecnologias Open Source*. [Internet] Faculdade de Engenharia da Universidade do Porto. Available from <http://repositorio-aberto.up.pt/handle/10216/71364> [Accessed 24th November 2014].

Oketunji, T. (2011) Design of Data Warehouse and Business Intelligence System. (June). Available from [http://medieteknik.bth.se/fou/cuppsats.nsf/all/812f5660b5f65276c125796a0064799c/\\$file/BTH2011Oketunji.pdf](http://medieteknik.bth.se/fou/cuppsats.nsf/all/812f5660b5f65276c125796a0064799c/$file/BTH2011Oketunji.pdf) [Accessed 27th November 2014].

OLAP.com (2015) *olap.com*. [Internet] Available from <http://olap.com/learn-bi-olap/olap-bi-definitions/business-intelligence/> [Accessed 7th August 2015].

Oracle (2015) *Indexing and Querying Spatial Data*. [Internet] Available from http://docs.oracle.com/cd/B19306_01/appdev.102/b14255/sdo_index_query.htm [Accessed 15th September 2015].

Oriani, F. (2014) *Difference between client API and server API*. [Internet] Available from <http://stackoverflow.com/questions/25405966/difference-between-client-api-and-server-api> [Accessed 8th April 2015].

Oueslati, W. and Akaichi, J. (2010) Mobile Information Collectors Trajectory Data Warehouse Design. *International Journal of Managing Information Technology* [Internet], 2 (3), pp.1–20. Available from <http://www.airccse.org/journal/ijmit/papers/0810ijmit01.pdf>.

Pelekis, N. and Frentzos, E. (2008) HERMES: aggregative LBS via a trajectory DB engine. *Proceedings of the 2008 ...* [Internet], p.1255. Available from <http://portal.acm.org/citation.cfm?doid=1376616.1376748&nhhttp://dl.acm.org/citation.cfm?id=1376748>.

Pentaho (2015) *Pentaho Dashboard*. [Internet] Available from <http://wiki.pentaho.com/dashboard.action> [Accessed 7th August 2015].

Pentaho PDI (2015) *Pentaho Data Integration*. [Internet] Available from https://www.pentaho.com/sites/default/files/uploads/resources/data_integration.pdf [Accessed 7th September 2015].

PentahoWiki (2010) *Pentaho Data Integration Steps*. [Internet] Available from <http://wiki.pentaho.com/display/EAI/Pentaho+Data+Integration+Steps> [Accessed 28th August 2015].

2015].

Pochampalli, S. (2015) *Pentaho BI Suite Tutorials*. [Internet] Available from <http://pentaho-bi-suite.blogspot.pt/> [Accessed 15th September 2015].

PostgreSQL (2015) *PostgreSQL*. [Internet] Available from <http://www.postgresql.org/> [Accessed 18th May 2015].

Power, D. (2007) *A Brief History of Decision Support Systems*. [Internet] Available from <http://dssresources.com/history/dsshhistory.html> [Accessed 6th August 2015].

Providence College (2014) *Overview of PivotTables*. [Internet] Available from http://www.providence.edu/it/Documents/PivotTables_Charts.pdf [Accessed 13th September 2015].

Ramazzina, S. (2013) *A first look to the new Pentaho BA Server 5.0 CE*. [Internet] Available from <http://ramathoughts.blogspot.de/2013/09/a-first-look-to-new-pentaho-bi-server-ce.html> [Accessed 10th August 2015].

República (2014) *Decreto-Lei nº 37/2014 de 14 de Março do Ministério da Economia*. [Internet] Available from http://www.imtt.pt/sites/IMTT/Portugues/Condutores/CartaConducao/Motociclos/Documents/DL37_2014.pdf [Accessed 10th September 2015].

Ribeiro, V., Rodrigues, J. and Aguiar, A. (2013) Mining geographic data for fuel consumption estimation. In: *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*. [Internet] IEEE, pp.124–129. Available from <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6728221>.

Ross, M. (2013) *Design Tip #152 Slowly Changing Dimension Types 0, 4, 5, 6 and 7*. [Internet] Available from <http://www.kimballgroup.com/2013/02/design-tip-152-slowly-changing-dimension-types-0-4-5-6-7/> [Accessed 7th August 2015].

Savidge, J. (2000) *Company Table*. [Internet] Available from <http://www.allergyfreefood.org/DBDocs/CompanyTable.html> [Accessed 18th August 2015].

Silva, A.M. (2012) *Um pouco de história: Business Intelligence*. [Internet] Available from http://www.seucurso.com.br/index.php?option=com_content&view=article&id=82:um-pouco-de-historia-business-intelligence-bi&catid=37:artigos&Itemid=27 [Accessed 6th August 2015].

Skog, I. and Handel, P. (2014) Indirect Instantaneous Car-Fuel Consumption Measurements. *IEEE Transactions on Instrumentation and Measurement* [Internet], 63 (12), pp.3190–3198. Available from <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6803061>.

SQLite (2015) *SQLite*. [Internet] Available from <https://www.sqlite.org/> [Accessed 18th September 2015].

Staff (2012) *10 Metrics to Optimize Fleet Efficiency*. [Internet] Available from

<http://www.automotive-fleet.com/article/story/2012/05/10-metrics-to-optimize-fleet-efficiency.aspx> [Accessed 15th August 2015].

Surendar, S. (2014) *Business intelligence- Components, Tools, Need and Applications*. [Internet] Available from <http://pt.slideshare.net/rajzeest/business-intelligence-components-tools-need-and-appl> [Accessed 20th June 2015].

Taylor, T. (2014) *What You Need to Know about Designing Information Dashboards*. [Internet] Available from <http://speckyboy.com/2014/10/30/designing-information-dashboards/> [Accessed 13th September 2015].

Team (2015) *Best Open Source Business Intelligence Tools For Tech Savvy Companies*. [Internet] Available from <https://www.yurbi.com/blog/best-open-source-business-intelligence-tools-for-tech-savvy-companies/> [Accessed 15th September 2015].

TechNet (2015) *Differences Between OLAP, ROLAP, MOLAP, and HOLAP*. [Internet] Available from <http://social.technet.microsoft.com/wiki/contents/articles/19898.aspx> [Accessed 28th July 2015].

Techopedia (2015a) *MySQL*. [Internet] Available from <https://www.techopedia.com/definition/3498/mysql> [Accessed 14th September 2015].

Techopedia (2015b) *PostgreSQL*. [Internet] Available from <https://www.techopedia.com/definition/3499/postgresql> [Accessed 15th September 2015].

Techopedia (2015c) *SQLite*. [Internet] Available from <https://www.techopedia.com/definition/24610/sqlite> [Accessed 14th September 2015].

Tezer, O. (2014) *SQLite vs MySQL vs PostgreSQL: A Comparison Of Relational Database Management Systems*. [Internet] Available from <https://www.digitalocean.com/community/tutorials/sqlite-vs-mysql-vs-postgresql-a-comparison-of-relational-database-management-systems> [Accessed 7th August 2015].

TrustRadius (2014) *The Best Full-Stack BI Software*. [Internet] Available from <https://www.trustradius.com/guides/business-intelligence/2014/best-software-fullstack> [Accessed 20th September 2015].

Webdetails (2015) *Webdetails - Community Dashboard Editor*. [Internet] Available from <http://www.webdetails.pt/ctools/cde/> [Accessed 10th August 2015].

Williams, B. (2010) *Data Model for Products, Orders and Deliveries*. [Internet] Available from http://www.databaseanswers.org/data_models/products_orders_and_deliveries/index.htm [Accessed 18th August 2015].

Williams, P. (2012) *history data warehouse*. [Internet] Available from <http://www.dataversity.net/a-short-history-of-data-warehousing/> [Accessed 6th August 2012].

Wood, S. (2007) *Pentaho - Mondrian Documentation*. [Internet] Available from <http://mondrian.pentaho.com/documentation/workbench.php> [Accessed 10th August 2015].