UNIVERSITÀ DEGLI STUDI DI SALERNO

SCUOLA DI DOTTORATO

PhD in Scienze e tecnologie dell'informazione, dei sistemi complessi e dell'ambiente – XIII ciclo

Tesi di Dottorato

Enhancing Data Warehouse management through semi-automatic data integration and complex graph generation

Mădălina Georgeta Ciobanu

Tutor
Prof.ssa Genny Tortora

Coordinatore del corso di dottorato
Prof. Roberto Scarpa

Maggio 2018

To my family

"Understand well as I may,
my comprehension can only be
an infinitesimal fraction
of all I want to understand."
-Ada Lovelace-

Summary

Strategic information is one of the main assets for many organizations and, in the next future, it will become increasingly more important to enable the decision-makers answer questions about their business, such as how to increase their profitability. A proper decision-making process is benefited by information that is frequently scattered among several heterogeneous databases. Such databases may come from several organization systems and even from external sources. As a result, organization managers have to deal with the issue of integrating several databases from independent data sources containing semantic differences and no specific or canonical concept description.

Data Warehouse Systems were born to integrate such kind of heterogeneous data in order to be successively extracted and analyzed according to the manager's needs and business plans.

Besides being difficult and onerous to design, integrate and build, Data Warehouse Systems present another issue related to the difficulty to represent multidimensional information typical of the result of OLAP operations, such as aggregations on data cubes, extraction of sub-cubes or rotations of the data axis, through easy to understand views.

In this thesis, I present a visual language based on a logic paradigm, named Complexity Design (CoDe) language, and propose a semi-automatic approach to support the manager in the automatic generation of a Data Warehouse answering

to his/her specific requirements. In particular, the manager expresses his questions in natural language and selects the needed information among different data sources. The selected data are imported from the databases, a data tuning process is enacted, and an association rule mining algorithm is run to extract the conceptual model underneath the data. Successively, the CoDe models are provided and the Data Mart is generated from the models. Finally, the CoDe model and the DW data are used to generate a graphical report of the required information. To shows the information the manager needs to make his decision according to the strategic questions, the report adopts Complex Graphs.

In this thesis, I also evaluate the effectiveness of the proposed approach, in terms of comprehensibility of the produced visual representation of the data extracted from the data warehouse. In particular, an empirical evaluation has been designed and conducted to assess the comprehension of graphical representations to enable instantaneous and informed decisions. The study was conducted at the University of Salerno, Italy, and involved 47 participants from the Computer Science Master degree having management, information systems and advanced database systems and data warehouse) knowledge. Participants were asked to comprehend the semantic of data using traditional dashboard and complex graph diagrams. The effort required to answer an evaluation questionnaire has been assessed together with the comprehension of the data representations and the outcome has been presented and discussed.

The achieved results suggest that people reached a higher comprehension when using Complex Graphs like the ones produced by our approach as compared to standard Dashboard Graphs. Furthermore, the analysis of the time needed to comprehend the data semantic shows that the participants spent significantly less time to understand the representation adopting a Complex Graphs visualization as compared to the standard Dashboard Graphs.

Both skilled and inexperienced users took advantages from the complex graph

representation in term of comprehension and effort, with most skilled participants taking a greater advantage in comprehension time. This finding could be very relevant for the decision maker. Indeed, complex graphs represent an effective visualization approach that enables a quicker and higher comprehension of data, improving the appropriateness of the decisions and reducing the decision making effort.

Based on this result, an automatic generation of data warehouse has been presented that uses complex graphs to visualize the main facts that a decision maker should use for strategic decisions. In particular, the CoDe language has been adopter to represent such information.

When assessing the automatic generation of data warehouse, the most critical phase of the process has been the data integration, due to the need of an important contribution from the manager. So, to encourage the implementation of the process in a real setting, a semi-automatic data integration process has been proposed and tested on 4 case studies to assess the goodness of the approach. The results, indicate that, despite some limitations, in three of the four case studies we obtained encouraging results.

Acknowledgements

I am grateful to my supervisor Professor Genny Tortora because this thesis would not have been possible without her support, understanding, patience and encouragement and for pushing me farther than I thought I could go.

It is a pleasure to thank Dr. Michele Risi for his great support during these years.

Finally, I would like to acknowledge with gratitude, the support and love of my family for believing in me. A big thank you goes to my mother who allowed me to get where I am and a special thanks to my husband Fausto for his patience.

I thank all the people dear to me, my old and new friends and the Marconi's management staff.

I dedicate this thesis to my children, Elisa and Rosario, who are my life, my joy, my everything and I hope one day they will live experiences like those of mum and dad and even better!

Contents

Summary III

Acknowledgements VII

1 Introduction 1
1.1 Thesis organisation . . . 4

2 Related Work 5
2.1 Data warehouse conceptual design . . . 5
2.2 The evaluation of graph . . . 7

3 Complex Graph modeling with CoDe 11
3.1 CoDe language overview . . . 14
3.2 The Complex Graph Generation tool . . . 20

4 Comprehension of Complex Graphs 25
4.1 Design of the Empirical Evaluation . . . 25
4.1.1 Context Selection . . . 25
4.1.2 Variable Selection . . . 26
4.1.3 Hypothesis Formulation . . . 28
4.1.4 Experiment Design, Material, and Procedure . . . 29
4.1.5 Analysis Procedure . . . 31
4.2 Results of the Empirical Evaluation . . . 33
4.2.1 Effect of Method . . . 33
4.2.2 Analysis of Cofactors . . . 35
4.2.2.1 Effect of Subjects Ability . . . 35
4.2.2.2 Effect of the Section . . . 35
4.2.3 Post-Questionnaire Results . . . 35
4.3 Discussion . . . 39
4.3.1 Implications . . . 39
4.3.2 Threats to Validity . . . 39

5 Automatic Generation of Data Warehouse 43
5.1 The proposed approach . . . 43
5.2 Strategic Question Definition . . . 47
5.3 Data Source Selection . . . 48
5.4 Concept Identification . . . 48
5.5 Complex Graphs Generated by CoDe . . . 52
5.6 CoDe Modeling . . . 56
5.7 Data Mart Generation . . . 60
5.7.1 Data Model Generation . . . 60
5.7.2 Logical/Physical Model Generation . . . 67
5.7.3 Data integration . . . 67
5.8 Reporting Tool . . . 70

6 Semi Automatic Data Integration 73
6.1 Semiautomatic Data Integration Process . . . 73
6.2 Integration process: from data sources to OLAP schema . . . 75
6.3 Data extraction and integration and construction of reconciled scheme 76
6.3.1 Data Sources . . . 76
6.3.2 Data preprocessing . . . 77
6.3.2.1 Preprocessing Data in R . . . 81
6.3.3 Syntactic analysis . . . 82
6.3.3.1 Computation of the Levenshtein distance . . . 82
6.3.3.2 Cluster analysis . . . 82
6.3.3.3 Advantages and disadvantages . . . 84
6.3.3.4 Syntactic analysis implementaion . . . 85
6.3.4 Semantic analysis . . . 87
6.3.4.1 Advantages and disadvantages . . . 88
6.3.4.2 Semantic analysis implementation . . . 89
6.3.5 Comparison of analysis results . . . 93
6.3.6 Data cleansing . . . 94
6.3.7 Data source Integration . . . 98
6.3.8 Reconciliation scheme . . . 100
6.3.9 A special case . . . 101
6.4 OLAP Schema Generation . . . 103
6.4.1 Conceptual design of the Data Warehouse . . . 103
6.4.1.1 Selection of the facts . . . 104
6.4.1.2 Attribute tree construction . . . 108
6.4.1.3 Definition of dimensions and measures . . . 108
6.4.1.4 Creation of the fact scheme . . . 110
6.4.2 Logical design of the Data Warehouse . . . 111
6.4.3 Implementation of the Data Warehouse . . . 114
6.4.3.1 SQL database . . . 115

6.4.3.2 Mondrian schema . . . 115
6.4.3.3 Population of the Data Warehouse . . . 119
6.5 OLAP analysis of the cube and strategic reports . . . 121
6.5.1 Multi-dimensional scheme queries . . . 122

7 Case studies 127
7.1 First case study: PANDA . . . 128
7.1.1 Data preprocessing . . . 129
7.1.2 Syntactic and semantic analysis . . . 130
7.1.3 Comparison of analysis results . . . 131
7.1.4 Source Integration . . . 132
7.2 Second case study: Vivaio . . . 132
7.2.1 Data preprocessing . . . 134
7.2.2 Syntactic and semantic analysis . . . 134
7.3 Third case Study: Ricette . . . 136
7.3.1 Data preprocessing . . . 137
7.3.2 Syntactic and semantic analysis . . . 137
7.4 Results . . . 138

8 Conclusion 141

Bibliography 145

List of Tables

4.1 Experimental Design 29
4.2 An excerpt of the Comprehension Questionnaire 30
4.3 PostExperiment Questionnaire 31
4.4 Likert scale adopted during the post-experiment questionnaire . . . 31
4.5 Descriptive statistics for Dependent Variables 33
4.6 Results of the analyses on method 35
5.1 Output of the Apriori algorithm in Weka 52
5.2 A comparison between extraction rules algorithms 54
6.1 Corpus documents fro the example case study 86
6.2 k-means execution for syntactic analysis 86
6.3 Correspondence matrix for tables S and T 97
6.4 Attributes used to identify fact measures 111
7.1 PANDA case study: k-means execution for syntactic analysis . . . 131
7.2 PANDA case study: k-means execution for semantic analysis . . . 131
7.3 Vivaio case study: k-means execution for syntactic analysis 135
7.4 Vivaio case study: k-means execution for semantic analysis 135
7.5 Ricette case study: k-means execution for syntactic analysis 137
7.6 Ricette case study: k-means execution for semantic analysis 137
7.7 Case studies statistics 139

List of Figures

3.1 Examples of dashboard diagrams . . . 12
3.2 Example of a complex graph . . . 13
3.3 The SUM function and its representation . . . 15
3.4 The NEST function and its representation . . . 16
3.5 Example of a complex graph . . . 16
3.6 Terms and functions of the CoDe notation . . . 17
3.7 The AGGREGATION function . . . 17
3.8 The AGGREGATION function corresponding graph . . . 18
3.9 The LINK relation in Code . . . 19
3.10 Visualization of the LINK relation . . . 19
3.11 The CoDe model . . . 20
3.12 The Complex Graph Generation tool architecture . . . 21
4.1 BoxPlots of the empirical analysis . . . 34
4.2 Interaction plots of Comprehension (a) and Effort (b) for Ability level 36
4.3 Interaction plots of learning effects for Comprehension (a) and Effort (b) . . . 37
4.4 PostExperiment Questionnaire results . . . 38
5.1 Data Models associated to the Case Study . . . 44
5.2 CoDe Conceptual Model for the Case Study . . . 45
5.3 The overall process to generate a DW from the strategic questions 46
5.4 An example of extracted rules and relationships identified from the conceptual models . . . 49
5.5 The CoDe model . . . 60
5.6 Tree attributes for the fact *orders* and elimination of cycles . . . 63
5.7 Tree attributes for the fact *productlines* and elimination of cycles . 64
5.8 Scheme for orders . . . 65
5.9 Fact scheme for product lines . . . 66
5.10 The data integration transformation for the fact *Orders* . . . 68
5.11 Data Integration Engine example . . . 69
5.12 Graphical representation of the CoDe model generated . . . 71
6.1 Generation Process . . . 74

6.2 Flowchart of the data integration process . . . 76
6.3 m x n term-document matrix . . . 79
6.4 Snippet of the code to build the terms-documents matrix . . . 81
6.5 An example cluster dendrogram . . . 83
6.6 Code snippet to build the SOM network . . . 90
6.7 SOM lattice (64 unit SOM) . . . 91
6.8 Code snippet to derive the covariances matrix . . . 92
6.9 Semantic analysis dendrogram . . . 92
6.10 Sample correspondence matrix . . . 97
6.11 Integrated table. Common attributes are shown in blue, attributes in A not matching in B are shown in green, while attributes in B not matching in A are shown in red . . . 99
6.12 Schema integration . . . 100
6.13 Three-tier data warehouse architecture . . . 101
6.14 Reconciled schema . . . 102
6.15 Intefgration strategy in case of even (left side) or odd (right side) tables to integrate . . . 102
6.16 Similarity matrix of the sample case study . . . 107
6.17 Error curve . . . 107
6.18 Algorithm used to build the attribute tree . . . 109
6.19 Scheme for the *orders* fact . . . 112
6.20 Logical snowflake scheme for the *orders* fact . . . 114
6.21 SQL script fragment for the DW creation . . . 116
6.22 Snowflake schema . . . 117
6.23 XML file fragment for *orders_cube* . . . 118
6.24 Example of ETL transformation that populates the Data Warehouse 120
6.25 Visualization of the *orders_cube* cube schema . . . 122
6.26 Managed orders for each business area query . . . 123
6.27 Third business rule report . . . 124
6.28 Analysis of the sales trend between 2003 and 2005 . . . 125
6.29 Year 2003 orders . . . 125
6.30 Year 2005 orders . . . 126
6.31 Customer's costs during the last three years . . . 126
7.1 E/R diagrams of PANDA case study data sources . . . 129
7.2 E/R diagram of the Vivaio E-Commerce database . . . 133
7.3 E/R diagram of the Vivaio Interventi database . . . 133

Chapter 1

Introduction

In modern organizations strategic information is becoming increasingly important for their health and survival of the business [1]. *Data Warehouse Systems (DWSs)* provide the decision-makers answers to questions about their business and how to increase their profitability. For a proper decision-making, a DWS has to contain a vast amount of heterogeneous data coming from several organizational operational systems and external data sources [2]. However, it is very difficult and onerous to integrate several data from independent databases containing semantic differences and no specific or canonical concept description [3]. Moreover, the costs to develop a DW are usually high and the results are often unpredictable. Furthermore, it frequently happens that both customers and developers are dissatisfied with the results of a DW project because of a long delay in system delivery and as well as a large number of missing or inadequate requirements [4].

Due to the different competencies and behaviors of managers and system developers, the requirement elicitation activities of a DW are very onerous and wasteful. As a consequence, developers usually need to iterate the requirement elicitation phase several times to obtain a stable and reliable requirements specification.

This complexity is also increased by the awareness that not only DWSs should support actual information needs but also have to meet future, at present unknown,

requirements [5] that will cause their evolution. For these reasons, there is the need to improve the DW development process by empowering and supporting the DW end-user, i.e., the manager, during his work. In the last decade, a great effort has been devoted to the development of DWs, mainly focusing on the system design. In particular, given the data needs, the main addressed topic regards the necessity to define what is the logical structure of the DW. Even assuming that the conceptual objects can be determined, the question about which are the useful conceptual objects for a DW and how these have to be determined is still an open issue. To address this issue, we need an explicit Requirements Engineering (RE) phase within the DW development [6]. Similarly, the necessity to rapidly update DWSs is faced in several related works [7], [8], [9], but little attention has been paid to individuate a simple and rapid methodology for capturing and comprehension of DWSs requirements. Another issue to address when dealing with complex DWSs is the comprehensibility and legibility of the results of queries, that are often represented using graph notations.

As a first contribution, in this thesis I will address the latter issue, i.e., the user comprehension of complex graphs produced during the design of a data warehouse and the corresponding visualization of a complex query that a manager needs to extract from a data warehouse system. In particular, I focus on a special visual language based on a logic paradigm, named Complexity Design (CoDe) language [10].

A CoDe model provides a high-level cognitive map of the complex information underlying the data within a DW. Information is extracted in tabular form using

the OnLine Analytical Processing (OLAP) operations[1]. The resulting model represents data and their interrelationships through different graphs that are aggregated simultaneously. It is worth mentioning that the CoDe model enables the user to specify the relationships among data and is independent of the chart type that can be freely selected by the manager.

The first contribution is to evaluate the effectiveness of the proposed approach, specifically concerning the comprehension of the produced visual representation of the data extracted from the data warehouse. To this aim, an empirical evaluation has been designed and conducted to assess the comprehension of graphical representations as a support to instantaneous and informed decisions. The study was conducted at the University of Salerno, Italy, and involved 47 participants from the Computer Science Master degree having management, information systems and advanced database systems and data warehouse) knowledge. Participants were asked to comprehend the semantic of data using traditional dashboard and complex graph diagrams. The effort required to answer an evaluation questionnaire has been assessed together with the comprehension of the data representations and the outcome has been presented and discussed.

Based on this finding, in this thesis I describe a semi-automatic approach to support the manager in the definition of the strategic questions he wants to ask to the data, the selection of the data source, the concept identification, the modeling of the data warehouse with CoDe, the Data Mart generation, and the reporting of the answers to his queries.

The automatic generation of data warehouse uses complex graphs to visualize

[1]The most popular OLAP operations are: *Roll-up*, performs aggregation on a data cube; *Drill-down*, that is the opposite of roll-up, i.e. it summarizes data at a lower level of a dimension hierarchy; *Slice*, performs a selection on one of the dimensions of the given cube, resulting in a sub-cube; *Dice*, selects two or more dimensions from a given cube and provides a new sub-cube; *Pivot* (or rotate), rotates the data axis to view the data from different perspectives; *Drill-across* combines cubes that share one or more dimensions; *Drill-through* drills down to the bottom level of a data cube down to its back-end relational tables; and *Cross-tab* aggregates rows or columns.

the main facts that a decision maker should use for strategic decisions. However, when assessing the automatic generation of data warehouse, we noticed that the most critical phase of the process was the data integration. In fact, this phase needs of a major contribution from the manager.

Thus, as a further contribution, to encourage the implementation of the process in a real setting, a semi-automatic data integration process has been presented. Finally, the proposed process has been tested on 4 case studies to assess the goodness of the approach. The results, indicate that, in three of the four case studies, results are promising.

1.1 Thesis organisation

The rest of the thesis is organized as follows. In Chapter 2 the related work in the field of data warehouse is reported. Research concerning the evaluation of graph visualization through empirical evaluations is also presented. In Chapter 3 the CoDe language adopted to model complex graph is presented. An evaluation of the comprehension of the produced visual representation is reported and discussed in Chapter 4. In Chapter 5 the overall process of the proposed approach for the generation of complex graph using the CoDe modeling is outlined and illustrated by using a sample case study. A semi-automatic data integration process in presented in Chapter 6 and applied to some case studies in Chapter 7. Finally Chapter 8 concludes the thesis and gives indications for future works.

Chapter 2

Related Work

2.1 Data warehouse conceptual design

The state of the art related to the proposed research mainly concerns DW conceptual design, that aims at proving an implementation-independent and expressive conceptual schema for the DW. The schema is designed by considering both the user requirements and the structure of the source databases, following a specific conceptual model [11]. Conceptual modeling generally is based on a graphical notation that is easy to understand by both designers and users. Conceptual modeling of DWs has been investigating considering two main approaches [11]:

- Multidimensional modeling
- Modeling of ETL (Extraction-Transformation-Loading)

ELT processes extract data from heterogeneous operational data sources, perform data transformation (conversion, cleaning, etc.) and load them into the DW. In approach, this is split into:

- the selection of the data, among the available heterogeneous databases, needed to the manager to answer his/her query

- the source joining, aiming at joining the different sources to group the data in a unique target.

Trujillo and Luján-Mora [12] presented a conceptual model based on UML supporting the design of ETL processes.

A data flow approach that can be applied to model any data transformation, from OLAP to data mining, has been proposed by Pardillo *et al.* [13]. The data warehousing data flows have been deeply investigated in the literature [14], [15], [16],[17],[18],[19] and in [20], [21], [22], [23], [19], [20], [21], [22].

Vassiliadis *et al.* [14] propose a solution that enables efficient continuous data integration in data warehouses, while allowing OLAP execution simultaneously. The issues focused in their work concern the DW end of the system, concerning how to perform loading of ETL processes and the DWs data area usage to support continuous data integration. Simitsis *et al.* [15] use code based ETL to continuously load data into an active data warehouse. The authors present the corresponding procedures and tables triggers for the active data warehouse in SQL and PL/SQL. Anfurrutia *et al.* [16], propose the ARKTOS ETL tool, that aims at modeling and enacting practical ETL scenarios by providing explicit primitives to capture common tasks. Russell and Norvig [24] describe the design and the implementation of a meteorological data warehouse by using Microsoft SQL Server. In particular, the proposed systems create process of meteorological data warehousing based on SQL Server Analysis Services and use SQL Server Reporting Services to design, execute, and manage multidimensional data reports. Hsu *et al.* [17] apply a clustering analysis on OLAP reports to automatically discover the grouping knowledge between different OLAP reports. Su *et al.* [18] propose a technique to connect several different data sources and uses a statistical calculation process to integrate them. Wojciechowski *et al.* [25] present a web based data warehouse application designed and developed for data entry, data management as well as reporting purposes. The same goal is addressed by Kulkarni *et al.* [19], Habich *et al.* [20], and Lehner *et*

al. [21], who introduces specific reporting functions. Finally, Du and Li [22] discuss a novel and efficient information integration technology aiming at information isolated island, based on data warehouse theory and other existing systems. In particular, the approach realizes data analysis on the basis of integration through physical views and a self-definition sampling strategy.

By the analysis all the above mentioned works we can observe that no one use DW loading and schema extraction methodologies during the requirement analysis phase. Similarly, to the best of my knowledge, application of techniques and tools to relate and display information extracted from a data warehouse supporting the requirement definition is still missing. Finally, the proposed approach differs from the existing ones for its capability to give an intuitive graphical representation of the semantic relationships between the data, also considering visualizations that can involve more than one type of graph. In particular, the approach proposed in this thesis allows to compose, aggregate, and change the different visualizations in a way that is simple to understand also by non-expert users.

2.2 The evaluation of graph

In the scientific literature, several research works have been devoted to the evaluation of graph notations, measuring the effects of different design aspects on user performance and preferences. However, there is very few works that investigate how different graphs can be related and can affect the way managers interpret information.

In this direction, Levy *et al.* [26] investigate people's preferences for graphical displays in which two-dimensional data sets are rendered as dots, lines, or bars with or without three-dimensional depth cues. A questionnaire was used to assess participants' attitudes towards a set of graph formats. Schonlau and Peters [27] assess that adding the third dimension to a visual representation does not affect

the accuracy of the answers for the bar chart. Moreover, they find that it causes a small but significant negative effect for the pie chart. The evaluation was conducted proposing to the subjects a series of graphs through the web. Participants accessed the graph on the PC monitors and answered several questions to assess the graph recall.

Inbar *et al.* [28] compared minimalist and non-minimalist versions of bar and pie charts to assess the appealing of the notations. The evaluation was based on a questionnaire aiming at collecting the preferences expressed by the participants. Results revealed a clear preference for non-minimalist bar-graphs, suggesting low acceptance of minimalist design principles.

Bateman *et al.* [29] evaluated whether visual embellishments and chart junk cause interpretation problems to understand whether they could be removed from information charts. To this aim, they compared interpretation accuracy and long-term recall for plain and Holmes-style charts (minimalist and embellished, respectively). Results revealed that people's accuracy in describing the embellished charts was similar to plain charts, while that their recall after a two-to-three week gap was significantly better.

Quispela and Maes [30] investigated the user perceptions of newspaper graphics differing in construction type. They considered two different kinds of users, namely graphic design professionals and laypeople. Participants rated the attractiveness and clarity of data visualizations. Also, the comprehension of graphics based on two-alternative forced-choice tasks was assessed.

Santos *et al.* [31] propose a dashboard with visualizations of activity data. The aim of the dashboard is to enable students to reflect on their own activity and compare it with their peers. This dashboard applies different visualization techniques. Three evaluations have been performed with engineering students, and aimed at assess the user perceptions of the considered approach. However, no comparison with the use of traditional dashboard or tabular data has been

conducted.

Fish *et al.* [32] evaluated whether the adoption of syntactic constraints called Well Formedness Conditions (WFCs) in Eulero Diagrams reduces comprehension errors. The participants had to fill in a questionnaire showing several diagrams together with a number of related assertions. The comprehension was assessed considering the number of correct answers provided.

Chapter 3

Complex Graph modeling with CoDe

In this Chapter the Complexity Design (CoDe) language is presented and a sample real scenario is used to describe the notation. In particular, the basic concepts of the complex graphs will be presented and illustrated in terms of the syntactic rules offered by the CoDe graphical language.

In many situations managers take decisions considering data related to key performance indicators represented in terms of graphs, combining categorical (nominal) and quantitative data. These data are usually extracted through OLAP operations from data warehouses and are represented adopting dashboard (standard) graphs, such as bars and pie charts. In particular, a bar chart enables the comparison of the values of each category by examining the lengths of the bars. A pie chart enables the comparison of the proportions of each category to the whole, by observing the angles of the segments. Generally, different aspects of the same problem are represented through separated graphs. This approach does not explicitly visualize relationships between information contained in different reports.

On the other hand, information should be provided to the managers in an effective way, to enable them to take quick and efficient decisions. In particular,

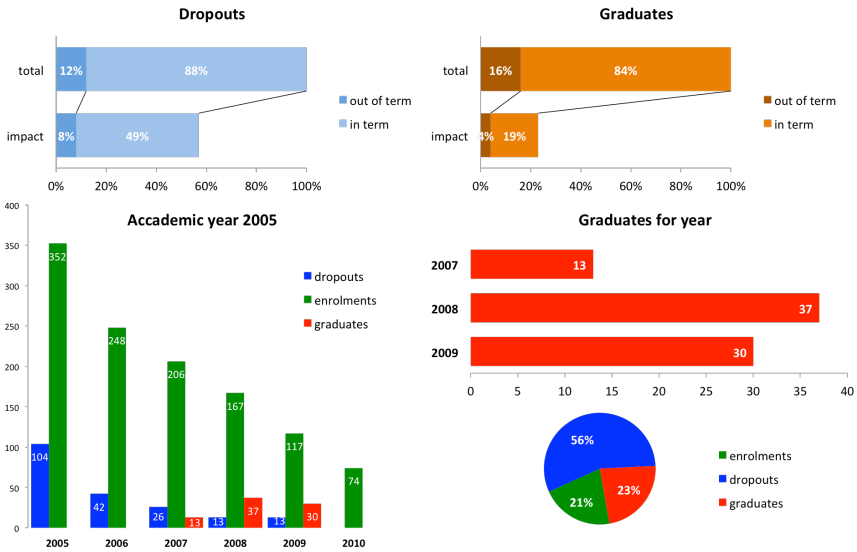


Figure 3.1. Examples of dashboard diagrams

following the suggestion by Bertin [33], the graph should answer the manager question with a single image. To this aim, we construct a *Complex Graph*, i.e., a graph connecting standard graphs through graphical relationships. With this approach, conceptual links between data become evident and the interpretability of the data should be improved.

The problem of formally describing complex graphs has been addressed by Risi *et al.* [34], where a logic paradigm to conceptually organize the visualization of reports, named CoDe language, has been proposed.

A CoDe model can be considered a high-level cognitive map of the complex information underlying the raw data stored into a data warehouse. Information is extracted in tabular form using the Online Analytical Processing (OLAP) operations. This model represents data and their interrelationships through different graphs that are aggregated simultaneously. It is worth mentioning that the CoDe model enables to specify the relationships among data and is independent on the

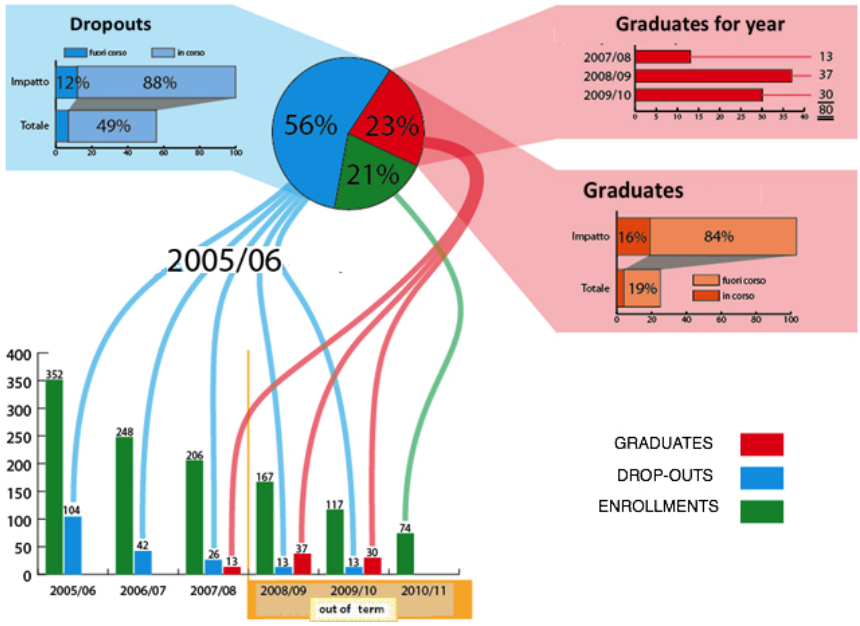


Figure 3.2. Example of a complex graph

chart type that can be freely selected by the manager. The complex graph generated by a CoDe model is also *well-formed*. The numerical proportion among data is maintained in related sub-diagrams, similarly to the different sectors of pie charts. Moreover, the disposition of the sub-graphs and their relationships has to avoid overlapping, whenever possible.

Figure 3.1 shows an example of data visualization of the students enrolled at the University of Salerno through the traditional dashboards diagrams, while Figure 3.2 depicts the complex graph generated from the CoDe model on the same data warehouse. The data source is composed of three data-marts concerning staff, students and accounting. In particular, the focus is on the students' careers, including information on: enrollments grouped by academic year, faculty, course of study, geographical source, gender, secondary school type and grade, and so on;

drop-outs; monitoring on exams and credits obtained per academic year, grades, duration of studies.

Figure 3.2 shows the trend of a Computer Science student cohort (i.e., all students enrolled in the 2005 academic year) subdivided into three categories (i.e., enrollments, graduates and dropouts) during six years after the enrollment to the Bachelor degree (the first three years concern the course degree period and the remaining the out-of-term period). In particular, the pie chart shows the status of the cohort 2005 in the year 2010 (56% of the students dropped out the study, 21% is still enrolled and 23% got the degree). The vertical bar chart shows the details on how these categories evolved during the considered six years. Links among the dropout sector of the pie depict how the 56% of the total number of student drop-out has been distributed along the considered time period. Moreover, the link between the enrollment sector of the pie and the enrollment bar of the histogram in the bottom left side of Figure 3.2 denotes that the enrollments represented in the pie are referred to ones of the year 2010. Similarly, the other charts provide further details.

3.1 CoDe language overview

In this Section, I resume the main CoDe rules and describe how they can be applied to model the students' enrollment complex graph in Figure 3.2. Further details about the CoDe language can be found in [34].

A standard graph depicts the information related to a data table. In the CoDe language a standard graph is described by a *Term*, represented by a rectangle and having n components. For example, Figure 3.6 shows the term *graduates*, with the six components {2005, 2006, ..., 2010}.

It is important to note that the CoDe model focuses on the abstraction of the information items and the relations between them rather than on the specific

shapes selected for the corresponding graphs and relationship links.

A complex graph is a graph connecting standard graphs represented by terms through the following relationships and compositions offered by the CoDe notation:

- SUM_i. Given two terms T_1 and T_2 where the value of i-th component in the second term is the sum of the components in the first one, the function SUM_i generates a complex graphs composed of the two terms T_1 and T_2 graphically related. An example of the SUM_i is provided in Figure 3.3.

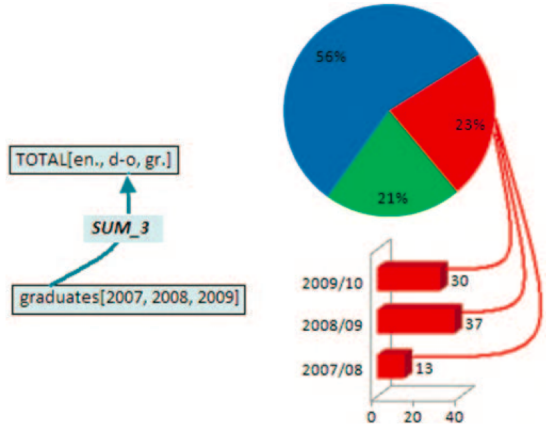


Figure 3.3. The SUM function and its representation

- $NEST\_i$. While SUM_i is an aggregation function, $NEST\_i$ disaggregates the i-th component into refined sub-categories. An example of $NEST\_i$ is provides in Figure 3.4

- $SHARE\_i$. Given the following terms: $T_{name}[C_1, \ldots, C_n], S_1[A_1, \ldots, A_i, A_{i+1}, \ldots, A_k], \ldots, S_n[Z_1, \ldots, Z_i, Z_{i+1}, \ldots, Z_h]$, the function $SHARE\_i$ constructs a complex graph by sharing all the n components of the report T_{name} on the i^{th} position of n reports $S_1, \ldots, S_n$, respectively.

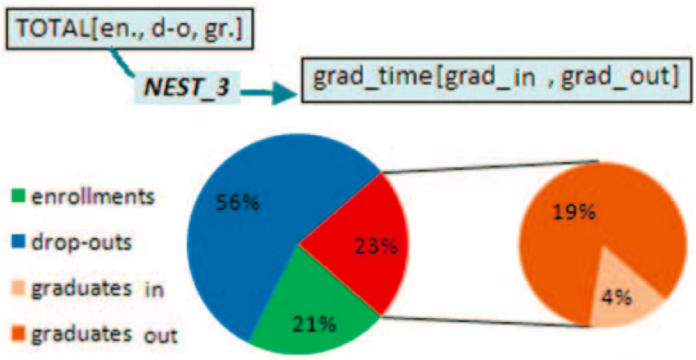


Figure 3.4. The NEST function and its representation

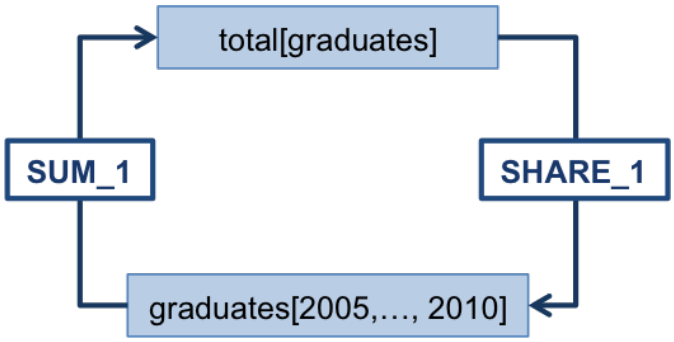


Figure 3.5. Example of a complex graph

As an example, Figure 3.5 shows the CoDe model of the complex graph in Fig. 3.6

- *AGGREGATION* is a function used to group several reports $T_1[C_1, \ldots, C_n], \ldots, T_k[C_1, \ldots, C_n]$ having the same components. They are grouped in a single term preserving the distinct identities of the related data series. The output term includes both the involved terms and the *AGGR* label which denotes the applied function. An example is shown in Fig. 3.7 and its graphical representation corresponds to the pie chart, vertical bar

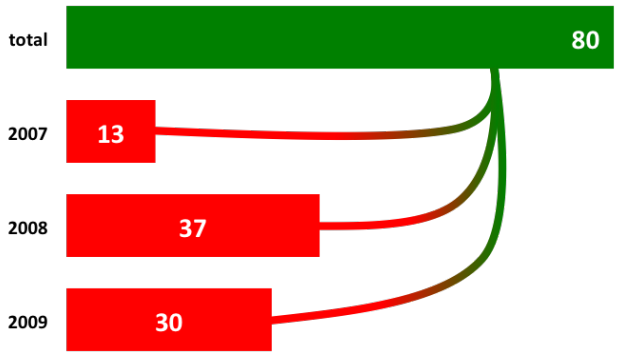


Figure 3.6. Terms and functions of the CoDe notation

chart and their links in Fig. 3.8.

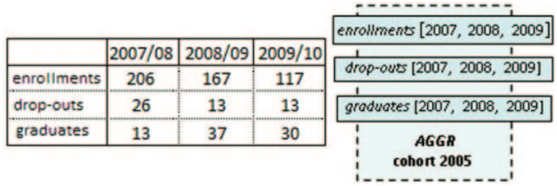


Figure 3.7. The AGGREGATION function

- $EQUAL\_i\_j$. This function relates the i-th component of term $T_1[D_1, \ldots, D_i, \ldots, D_h]$ with the j-th component of term $T_2[C_1, \ldots, C_j, \ldots, C_n]$. In particular, these two component are equal and they have the same associated value.

- The $PLUS$ operator adds graphic symbols and the $ICON$ operator allows to metaphorically visualize input components with suitable icons.

  As an example, these operators allow to add vertical or horizontal lines with labels or represent data with bitmaps that are proportionally related to the

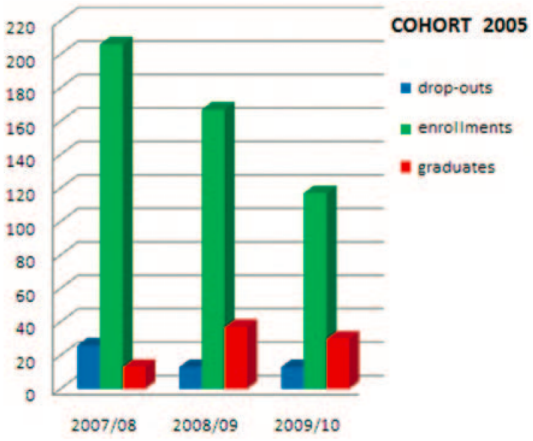


Figure 3.8. The AGGREGATION function corresponding graph

component values associated to them. These operators can be used both to improve the aesthetic impact of the final visualization and to add auxiliary information not given in the represented raw-data.

- $LINK$. This relation states the existence of a logical relationship between terms.
  The $LINK$ relation between input reports $T_1[C_1, \ldots, C_n]$ and $T_2[D_1, \ldots, D_k]$ generates a complex graph composed of the two terms T_1 and T_2 whose each couple of components (C_i, D_j) $\forall i, j$ is graphically related if there exists a valid value among them. Figure 3.9 illustrates the $LINK$ relation between the complex terms and the further terms corresponding to the considered impact reports (i.e., grad IMPACT and drop outs IMPACT terms). A graphical representation of the CoDe model is shown in Figure 3.10

The complete CoDe model of the students' enrollment complex graph is depicted in Fig. 3.11.

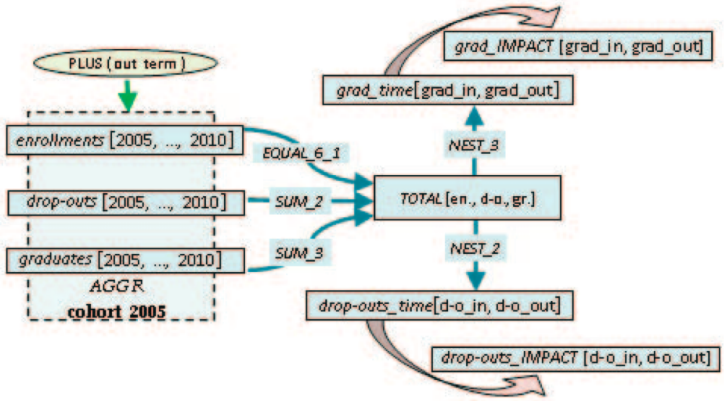


Figure 3.9. The LINK relation in Code

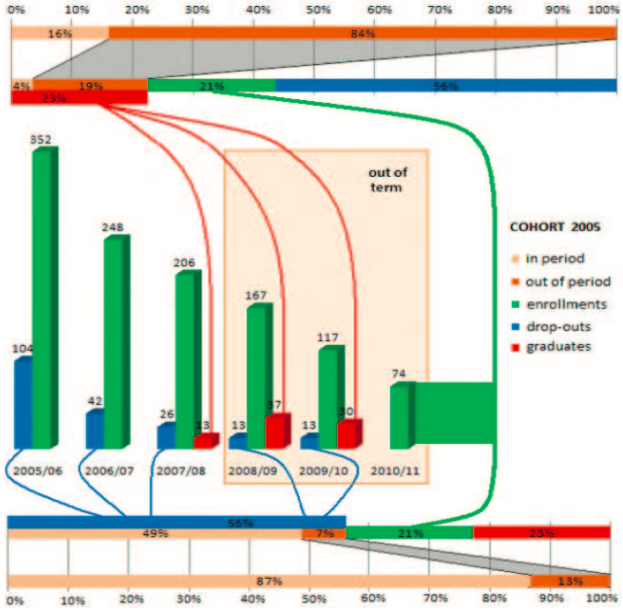


Figure 3.10. Visualization of the LINK relation

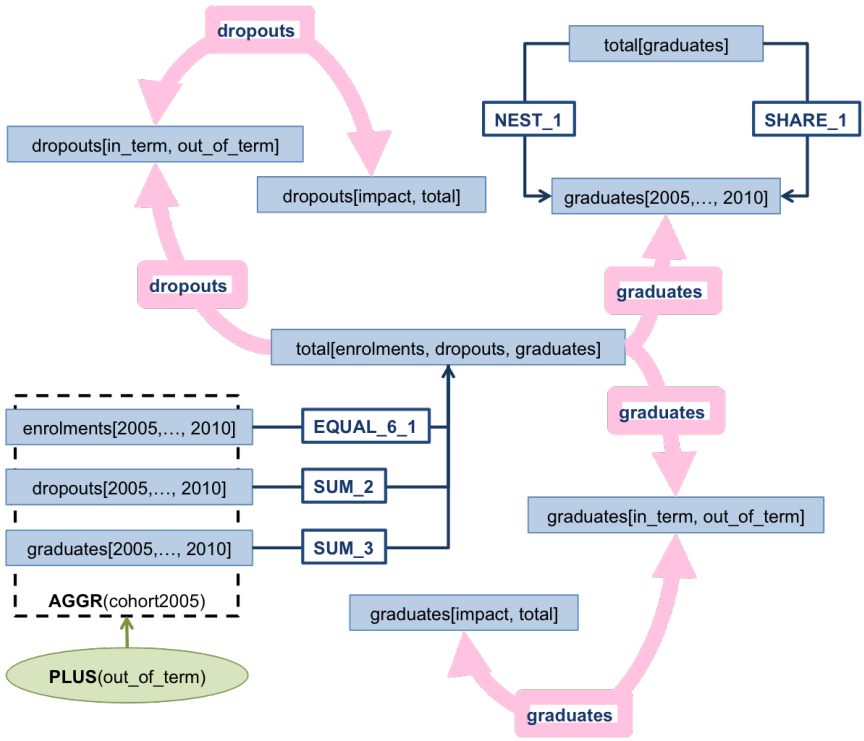


Figure 3.11. The CoDe model

3.2 The Complex Graph Generation tool

A graphical tool to model and visualize the complex graphs has been developed as an Eclipse plug-in [34]. In this thesis the focus is on the process to generate and visualize complex graphs through the Web browser. In the future it will be possible to improve the user interaction by adding new features, such as animations and handling more sophisticated interactions (e.g., overlays, tooltip, data information). In the following, details will be provided concerning the structuring of complex graph layout taking into account the size and the numerical proportion among data, and trying to avoid overlapping. The graphical tool architecture, depicted in Fig. 3.12, consists of the following components:

- The *CoDe Editor* is a graphical environment that enables the manager to design the CoDe model, the visualization layout of the complex graph including the standard graph types, and other visualization details.
- The *Data Manager* accesses to the DW schema and provides the CoDe Editor with the terms needed to design a CoDe model. The CoDe Editor outputs the XML description of the CoDe model. In particular, the XML provides information about the logical coordinates (i.e., the xy-position of the term in the editing area of the CoDe model, the ratio scale, the standard graph type associated to each term in the CoDe model, and the OLAP query to extract data from DW.
- The *Report Generator* takes the XML description in input and uses the OLAP queries to obtain the multidimensional data from the Data Manager. The Report Generator converts these data into a JSON format and saves them into a file. Moreover, it creates a file containing the rendering procedure (coded in Javascript) for drawing the complex graph. To this aim, it exploits the features (i.e., graphical primitives and charts) of the *Graphic Library*. An HTML file is also created. All these files are provided to the browser (i.e., the Viewer) that loads the HTML file and executes a rendering procedure to visualize the complex graph starting from the data file.

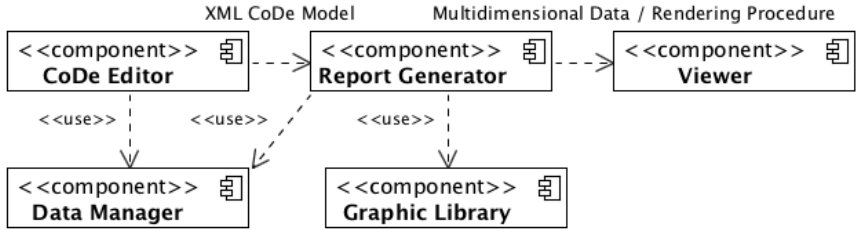


Figure 3.12. The Complex Graph Generation tool architecture

In the following, we describe the process to generate the rendering procedure.

The procedure draws the charts and the relations between them performing the following 6 steps.

1. For each term in the CoDe model, compute the size of each single graph exploiting data extracted by the OLAP query and the ratio scale associated to it. To this aim the key-points of a chart are computed (e.g., the vertexes of a bar in an histogram) in order to represent its sub-parts. These points are used to determine the bounding box of the graph.

   a. Each AGGREGATION function in the CoDe model is considered as a single term.

   b. If a SHARE function is applied among two terms, the shared sub-part is added to the chart of the second term. The size and the bounding box are updated accordingly.

   c. The bounding boxes are arranged taking into account the logical coordinates of the terms in the CoDe model.

2. For each relation EQUAL, SUM or NEST between two terms in the CoDe model, the dimensions of the charts representing the terms are computed as follows:

   a. A ratio between the charts is computed by considering the areas of the sub-parts involved in the relation. In particular, the area of the chart in the right side of the relation becomes equivalent to the area of the chart in the left side of the relation. If different ratios are computed for the same chart, the ratio relative to the EQUAL relation is considered, otherwise the average ratio value is used.

   b. The bounding boxes and the key-points are resized by considering the computed ratios.

3. If overlapping charts exist, the center of the complex graph is computed taking into account all the bounding boxes. All the charts are moved along the radial direction with respect to the center until charts do not overlap anymore.

4. The charts are drawn.

5. The relations among the related sub-parts (i.e., EQUAL, NEST, SUM, and LINK) are drawn by adopting B-spline functions on the key-points of the charts. In particular, a path detection algorithm avoids (or minimizes) the overlap among charts or other relations.

6. Graphical symbols and icons (specified for the PLUS and ICON operators) are added to the complex graph taking into account the terms they are related to.

Chapter 4

Comprehension of Complex Graphs

This chapter provides the design and planning of the experiment, structured according to the guidelines of Juristo and Moreno [35], Kitchenham *et al.* [36], and Wohlin *et al.* [37]. The results of the empirical evaluation are also presented end discussed.

4.1 Design of the Empirical Evaluation

Applying the Goal Question Metric (GQM) template [38], the goal can be defined as: *analyze* dashboard and complex graph *for the purpose* of evaluating their use *with respect to* the *comprehension* of data correlation *from the point of view of* a decision maker *in the context of* university students with management knowledge and real data warehouses.

4.1.1 Context Selection

The context of the experiment is the comprehension of graphical representations to enable instantaneous and informed decisions. The study was executed at the

University of Salerno, Italy. Participants were 47 Computer Science Master students having passed the exams of Management, Information Systems and advanced Database Systems and Data Warehouse.

All of them were from the same class with a comparable background, but different abilities. Before conducting the experiment, the students were asked to fill in a pre-questionnaire aiming at collecting their demographic information. A quantitative assessment of the ability level of the students was obtained by considering the average grades obtained in the three exams cited above. In particular, students with average grades below a fixed threshold, i.e., 24/30[1] were classified as Low Ability (*Low*) subjects, while the other students were classified as High Ability (*High*) subjects. It is important to note that other criteria could be used to assess the subjects' ability. However, the focus of our experiments is on notations for data visualization, thus, considering the grades achieved by the students in courses on this topic where several dashboard diagrams are used is a reasonable choice.

The diagrams proposed during the evaluation are related to two datasets extracted from the following real data warehouses:

DW1. The data warehouse of the University of Salerno.

DW2. The Foodmart data warehouse, providing information about customers and sales of food products carried out in the various category of shops (supermarket, grocery stores, etc.) in the United States, Mexico and Canada between 1997 and 1998.

4.1.2 Variable Selection

We considered the following independent variables:

[1]In Italy, the exam grades are expressed as integers and assume values between 18 and 30. The lowest grade is 18, while the highest is 30.

Method: this variable indicates the factor the studies are focused on. Participants had to fill-in the evaluation questionnaire, composed of two sections, consisting in the comprehension of a graphic visualization of data. As we wanted to investigate the comprehension of the complex graph, the experiment foresaw two possible treatments, referred to two data representation views:

- *Dashboard Graph (DG).* Participants try to comprehend the data semantic represented using traditional Dashboard diagrams;
- *Complex Graphs (CG).* Participants try to comprehend the data semantic represented using complex graph diagrams.

To analyze the participants' performances, we selected the following two dependent variables:

- *Effort:* the time a participant took for fill in a section of the questionnaire;
- *Comprehension:* we asked the participants to answer a comprehension questionnaire, composed of two sections. The answers were assessed using an Information Retrieval metric.

  In particular, comprehension is evaluated in terms of *F-Measure* [39], adopting *Precision* and *Recall* measures to evaluate the comprehension level of data visualization techniques. Given:

  - n: the number of questions composing a given section of the questionnaire.
  - m_s: the number of answers provided by the participant s.
  - k_s: the number of correct answers provided by the participants s.

  we can compute the information retrieval metrics as follows:

$$precision_s = \frac{k_s}{m_s} \tag{4.1}$$

$$recall_s = \frac{k_s}{n} \tag{4.2}$$

Precision and recall measure two different concerns, namely correctness and completeness of the answers, respectively. To balance them, we adopted their harmonic mean

$$F - Measure_s = \frac{2 \cdot precision_s \cdot recall_s}{precision_s + recall_s} \tag{4.3}$$

This means has been used to measure the Comprehension variable. All the measures above assume values in the range [0,1].

Other factors may influence both the directly dependent variables or interact with the main factor For this reason, the following co-factors have been taken into account:

- *Ability.* The participants having a different ability could produce different results.
- *Section.* As better detailed in Section 4.1.4, the participants had to fill in two questionnaire sections. Although we designed the experiment to limit the learning effect, it is still important to investigate whether subjects perform differently across subsequent sections.

4.1.3 Hypothesis Formulation

The objective of our study is to investigate the effectiveness of complex graphs on the user comprehension. In particular, since complex graphs add graphical relationships and proportionality related to the data among the sub-graphs composing it, we are interested in investigating whether such additional details increase the comprehension level.

The *null hypotheses* the controlled experiment aimed at testing are:

- H_{n1}: there is no significant difference in the *Comprehension* when using *CG* or *DG*;
- H_{n2}: there is no significant difference in the *Effort* when comprehending a *CG* or a *DG*;

The hypothesis H_{n1} and H_{n2} are two-tailed because we did not expect a positive or a negative effect on the comprehension of the data correlation on the experiment tasks.

4.1.4 Experiment Design, Material, and Procedure

Participants were split into four groups, making sure that *High* and *Low* ability participants were equally distributed across groups. Each group was composed of participants receiving the same treatments in the questionnaire. The experiment design is of the type "*one factor with two treatments*", where the factor in this experiment is the graph type and the treatments are the *CG* and the *DG*. We assign the treatments so that each treatment has equal number of participants (balanced design). The experiment design is shown in Table 4.1.

Table 4.1. Experimental Design

	Group 1	**Group 2**	**Group 3**	**Group 4**
Section 1	T_1-DG	T_1-CG	T_2-DG	T_2-CG
Section 2	T_2-CG	T_2-DG	T_1-CG	T_1-DG

The design ensured that each participant worked on a different task in a single session, receiving a different treatment for each task. Also, the design allowed us to 1) consider different combinations of task and treatment in different order across the first two sections and 2) the use of statistical tests for studying the effect of multiple factors.

Before the experiments, subjects have been trained on the complex graph and

Table 4.2. An excerpt of the Comprehension Questionnaire

ID	Question
Q1	How many students enrolled in 2005?
Q2	Which is the percentage of students graduated outside the prescribed amount of time over the total amount of graduates?
Q3	How many students enrolled in 2010/11?
Q4	How many students dropped-out between 2005 and 2011?
Q5	How many students obtain the degree within the prescribed time?
Q6	Which is the percentage of students that obtain the degree within the prescribed time over the total amount of graduates?

dashboard graphic notations. The introduction to these notations required ten minutes.

During the experiment, we assigned the following two comprehension tasks to each participant, designed in such a way to reflect the typical analyses that a manager has to perform. In particular, participants had to fill in a comprehension questionnaire composed of 2 parts:

- T_1. Comprehension of graphs related to *DW1*;
- T_2. Comprehension of diagrams related to *DW2*.

In these two tasks the users were presented a graphical description of real data, extracted by the selected data warehouses by using OLAP operation. The coherence and completeness of the proposed graphs were validated by a data analysis expert of the University of Salerno. Each task proposed a number of related multiple-choice questions. Participants had to indicate the choices that they believed to be correct. The participants filled their questionnaire and annotated, for each section, the starting and ending time.

An example of the section associated to the task T_1 is reported in Table 4.2, related to the graphs shown in Figures 3.1 and 3.2 for the method *DG* and *CG*, respectively.

At the end of each section, participants annotated the ending time. Section have been completed sequentially. At the end of the experiment, the participants

Table 4.3. PostExperiment Questionnaire

ID	Question
P1	The questions of the evaluation questionnaire were clear to me.
P2	The goal of the experiment was clear to me.
P3	I found useful the relationships of the CG representation to comprehend data.
P4	The understanding of the data semantic was problematic when they are represented using CG.
P5	The proportion among sub-graphs of CG is useful to comprehend data.
P6	The arrangement of sub-graphs of CG helps to comprehend data.

Table 4.4. Likert scale adopted during the post-experiment questionnaire

1.Fully agree	2.Weakly agree	3.Neutral	4.Weakly disagree	5.Strongly disagree

filled in a post-experiment questionnaire (see Table 4.3). composed of 6 questions expecting closed answers scored using the following Likert scale [40] (see Table 4.4).

The purpose of the post-experiment questionnaire is to assess whether the questions and the experiment objectives were clear, and the gather the subject's opinion on the usefulness of the *CG* and its characteristics.

4.1.5 Analysis Procedure

To test the defined *null hypotheses*, parametric statistical tests have been used. To apply parametric test the normality of data is verified by means of the Shapiro & Wilk test [41]. This test is necessary to be sure that the sample is drawn from a normally distributed population. A p-value smaller than a specific threshold (α) allows us to reject the *null hypotheses* and conclude that the distribution is not normal. This normality test has been adopted because of its good power properties as compared to a wide range of alternative tests. In case the p-value returned by the Shapiro-Wilk test is larger than α, we used an unpaired t-test. Unpaired teats are exploited due to the experiment design (participants experimented *CG* and *DG* on two different experiment objects). In the statistical tests we decided to accept a probability of 5 percent of committing Type I errors, i.e., of rejecting the *null hypotheses* when it is actually true. Thus, the *null hypotheses* is rejected when the

appropriate statistical tests provide a p-value less than the standard α-level of 5 percent (0.05).

In addition to the tests for the hypotheses formulated in Section 4.1.2, we also evaluated the magnitude of performance difference achieved with the same user group with different methods. To this aim, we evaluated the effect size.

The effect size can be computed in several ways. In case of parametric analyses, we used *Cohen's d* [42] to measure the difference between two groups. It is worth noting that the effect size can be considered negligible for $d < 0.2$, small for $0.2 \leq d \leq 0.5$, medium for $0.5 \leq d \leq 0.8$, and large for $d > 0.8$ [43]. On the other hand, for non-parametric analysis we used the *Cliff's Delta* effect size (or d) [42]. In this case, according to the literature, the effect size is considered small for $d < 0.33$ (positive as well as negative values), medium for $0.33 < d < 0.474$ and large for $d > 0.474$.

For parametric tests, the statistical power (i.e., post hoc or retrospective power analysis) has been analyzed. To compute the power, the type of hypothesis tested (one-tailed vs. two-tailed) has been considered . Whatever is the kind of test, statistical power represents the probability that a test will reject a *null hypothesis* when it is actually false. Then, it is worthy to be analyzed only in case a *null hypothesis* is rejected. The statistical power value rage is [0,1] where 1 is the highest value and 0 is the lowest. Note that the higher the statistical power, the higher the probability to reject a *null hypothesis* when it is actually false. A value greater than or equals to 0.80 is considered as a standard for adequacy [44]

Finally, interaction plots have been used to study the presence of possible interactions between the main factor and the ability co-factor. Interaction plots are line graphs in which the means of the dependent variables for each level of one factor are plotted over all the levels of the second factor. If the lines are almost parallel no interaction is present, otherwise an interaction is present. Cross lines indicate a clear evidence of an interaction between factors.

Table 4.5. Descriptive statistics for Dependent Variables

Dependent variable	Method	median	mean	st. dev.	min	max
Comprehension	CG	0.83	0.83	0.16	0.33	1
	DG	0.83	0.78	0.17	0.36	1
Effort	CG	10	10.38	3.44	4	21
	DG	12	12.21	3.03	7	20

We show the distribution of the data of the post-experiment survey questionnaire using histograms. They provide a quick visual representation to summarize the data.

4.2 Results of the Empirical Evaluation

In the following subsections, the results of this experiment are reported and discussed.

4.2.1 Effect of Method

Firstly, the effect of Method on the dependent variables is analyzed. Table 4.5 reports descriptive statistics of the dependent variables Comprehension and Effort, grouped by treatment (*DG*, *CG*). The results achieved are also summarized as boxplots in Fig. 4.1. These statistics show that the average Comprehension was higher in case of *CG* (6% more on average), while users took on average 1 minute and 43 seconds (15%) less to comprehend the same problem when using a *DG* with respect to a *CG*.

By applying Shapiro-Wilk normality test to the F-measure *DG* and *CG* samples it results $p-value < 0.001$ for both the samples. Since all p-values are less than 0.05 the samples are not normally distributed and we have to adopt a non-parametric test. Applying the Wilcoxon Rank-Sum Test we determine the p-values reported in

Table 4.6. Concerning the Time performance, by applying Shapiro-Wilk normality test we get $p - value = 0.317$ and $p - value = 0.022$ for DG and CG, respectively. Since all p-values are greater than 0.05, both the samples are normally distributed and we can adopt a parametric test (t-test). Table 4.6 summarizes the results of the hypotheses tests for H_{n1} and H_{n2} and reports the values of effect size and statistical power. This table also shows the number of people that reached a higher Comprehension with CG and DG. Similarly, the results on Effort are reported.

The Wilcoxon Rank-Sum Test revealed that there exists a significant difference on comprehension in terms of Comprehension ($p - value = 0.045$) in case participants that used CG and DG (i.e., H_{n1} can be rejected). The effect size is small. Concerning H_{n2}, the Wilcoxon Rank-Sum Test revealed that it (i.e., the *null hypothesis* used to assess the influence of Method on Effort) can be rejected since $p - value = 0.007$. This value is highlighted in bold. The effect size was medium and negative (i.e., -0.564) and the statistical power was 0.772.

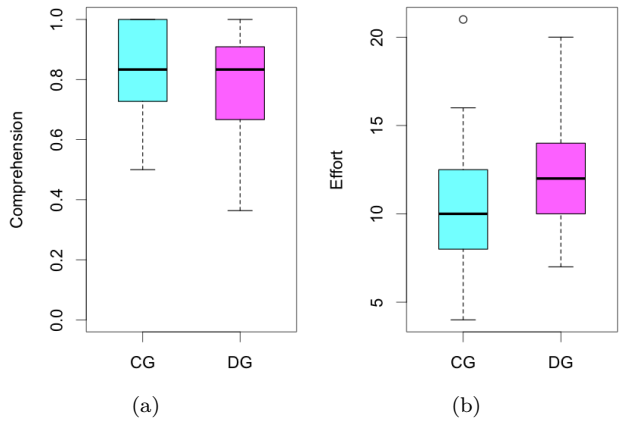


Figure 4.1. BoxPlots of the empirical analysis

Table 4.6. Results of the analyses on method

Hypotheses	p-value	Effect size	Stat. Power
H_{n1}(Comprehension)	**Yes (0.045)**	0.234 (Cliff's delta small)	-
H_{n2}(Effort)	**Yes (0.007)**	-0.564 (Cohen medium)	0.772

4.2.2 Analysis of Cofactors

4.2.2.1 Effect of Subjects Ability

The analysis of the interaction plots in Fig. 4.2(a) highlights that participants with *High* ability achieved better Comprehension than *Low* ability subjects in both the *CG* and *DG* treatments. The analysis also revealed that there is no interaction between Method and Ability: the gap remain the same with the two methods, with a performance improvement for *CG*.

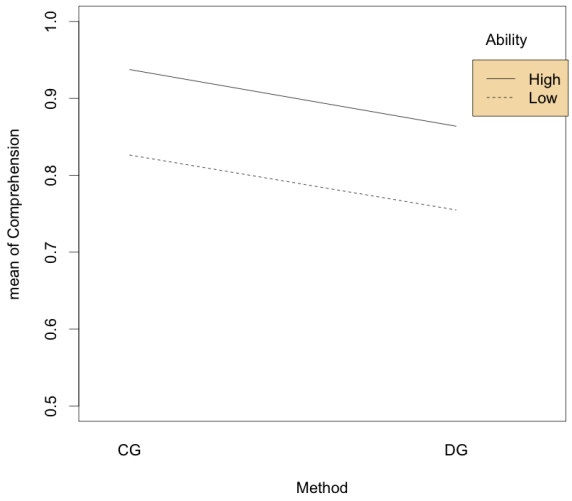
We also analysed the influence of the participants' Ability on Effort. Fig. 4.2(b) shows that the gap between *Low* and *High* Ability participants increments in favor of *CG*.

4.2.2.2 Effect of the Section

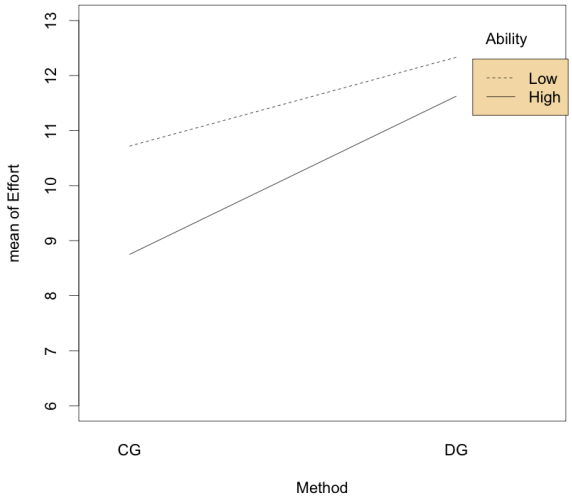
To investigate if there is a learning effect when the participants answered successive sections of the questionnaire, we analyzed the Interaction Plots in Fig. 4.3. In particular, it is possible to observe that for both Comprehension and Effort there is a learning effect. This is due to the task type (a comprehension task) that is similar in the two sections, except for the task domain and for the graph notation. However, this little learning effect has been mitigated by the balanced experiment design.

4.2.3 Post-Questionnaire Results

The data collected from the post-experiment survey questionnaires are visually summarized in Fig. 4.4.



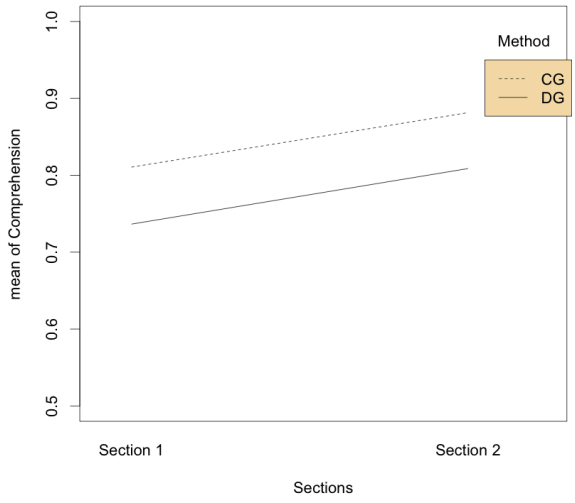
(a)



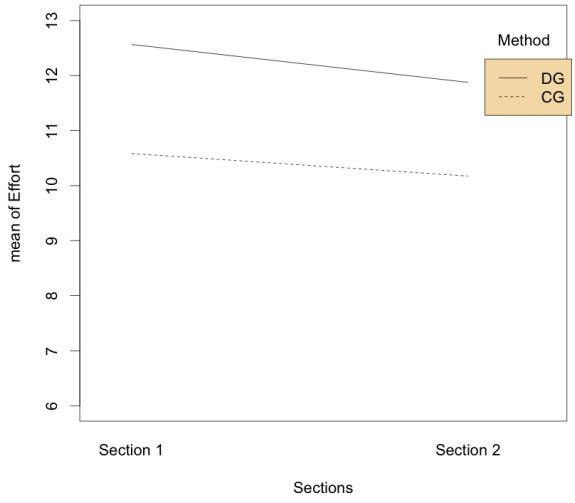
(b)

Figure 4.2. Interaction plots of Comprehension (a) and Effort (b) for Ability level

P1 suggests that many participants considered the questions of the comprehension tasks clear (10 expressed a very positive judgment and 29 a positive judgment).



(a)



(b)

Figure 4.3. Interaction plots of learning effects for Comprehension (a) and Effort (b)

The histogram of P2 shows that the participants found the objectives of the experiment clear (11 expressed a very positive judgment and 27 a positive judgment).

The participants found useful the relationships of the *CG* representation for comprehending data (P3), 14 expressed a very positive judgment and 22 a positive judgment. By looking at the answers to questions P4, many participants did not considered problematic understanding the semantic of data represented using *CG* (14 scored 5 and 22 scored 4). 11 participants considered the proportion among sub-graphs of *CG* very useful for comprehending data, while it was considered useful for 19 of them (P5). Concerning the help provided by the arrangement of sub-graphs of *CG* for comprehending data (P6), the participants were less satisfied. Indeed, 8 expressed a very positive judgment, while 17 expressed a negative.

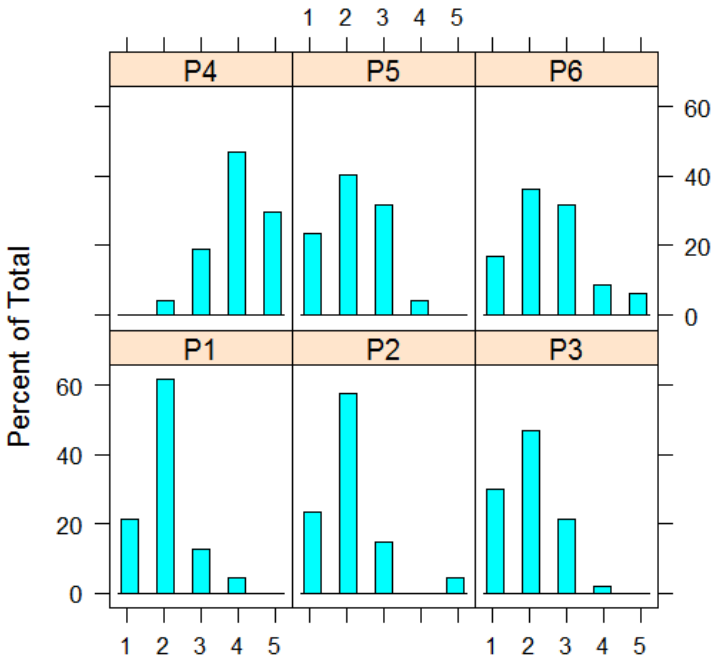


Figure 4.4. PostExperiment Questionnaire results

4.3 Discussion

In the following, the achieved results and their possible practical implications will be discussed. Obviously, a discussion on the threats that could affect the validity of our results is always necessary in this kind of empirical investigations.

4.3.1 Implications

The results of the data analysis conducted on the proposed experiment indicate a high average comprehension with both the two treatments and a significant difference between the comprehension of a *CG* and of a *DG* diagram, with a small effect size. Also, the results related to the comprehension time indicated that the participants spent significantly less time to understand a problem adopting a *CG*, with a medium effect size. This suggests that the two kinds of diagrams correctly describe the data, with a little advantage for *CG*, and that the comprehension requires less time in the *CG* case. Both *High* and *Low* ability users took advantages of the complex graph representation in term of both comprehension and effort. *High* ability participants performed better in both cases, but the gap increases when analyzing the effort. This suggests that *High* ability participants took a greater advantage in comprehension time. This finding is relevant for the decision maker. Indeed, complex graphs represent an effective visualization approach that enables a quicker and higher comprehension of data, improving the appropriateness of the decisions and reducing the decision making effort.

4.3.2 Threats to Validity

In this Section the threats to validity that could affect the study results will be discussed. In particular, we focus here on construct, internal, conclusion, and external validity threats.

Internal Validity threats concern factors that may affect our dependent variables and are relevant in studies that try to establish a causal relationship. They can be due to the learning effect experienced by subjects between sections. As shown in Section 4.2.2.2, there is a little learning effect between the two sections. To mitigated this effect, the experiment design has been balanced. In particular, subjects worked by permuting two sections, on different tasks and using two different graphical approaches. The fatigue effect is not relevant, since the experiment lasts at most 38 minutes and 22.6 minutes on average. Another issue concerns the possible information exchanged among the participants while performing the tasks. This threat has been prevented by observing participant during the questionnaire compiling. Finally, the participants did not know the hypotheses of the experiments and were not evaluated on their results.

Construct Validity threats concern the possibility that the relationship between cause and effect is causal. This validity was mitigated by the experiment design used. Selection and measurements of the dependent variables could threaten construct validity. Comprehension was measured using an information retrieval based approach. The comprehension of the questions were also assessed by the post-experiment questionnaire (see Table 4.3), whose results are provided in Figure 4.4. The collected answers revealed that the questions and the experiment objective were clear. Regarding the variable time, we asked the participants to note down the start and the stop time when they filled in a questionnaire section. This information was also validated by supervisors. The comprehension was assessed answering questions based on data related on real databases. In addition, the *DG* and *CG* graphs proposed in the experiment were validated by a data analysis expert for verifying whether the graphs were appropriate for answering the questionnaire requests.

Conclusion validity concerns the relationship between the treatment and the obtained findings. In is worth noting that the experiment design also mitigated

Conclusion Validity threats. In the conducted experiment, the selection of the population may have affected this experimental validity. To reject the *null hypotheses*, we used statistical tests and power analyses. In case differences were present but not significant, this was explicitly mentioned, analyzed, and discussed. The conclusion validity could be also affected by the sample size, consisting of 47 participants. To this aim, the statistical power of the tests was evaluated. Finally, the used post-experiment survey questionnaire was designed using standard metrics and scales [40].

External validity concerns the possibility of generalizing our findings within different contexts. This threat may present when experiments are conducted with students, since they could not be representative of decision makers. This threat is partially mitigated by the fact that students involved in the experiment passed specific exams of the computer science masters degree. Their ability to interpret dashboard graphs has been assessed, as also the f-measure performances revealed. In addition, working with students implies various advantages, such as the fact the students' prior knowledge is rather homogeneous, and there is the availability of a large number of participants [45]. Obviously, to increase our awareness in the achieved results, case studies within industrial contexts are needed. Other external validity threats are represented by the tasks assigned to the participants. The tasks we selected were taken from real databases. Thus, they are representative of the typical decision activities.

Chapter 5

Automatic Generation of Data Warehouse

In the previous Chapter, the results of an empirical study have been presented to demonstrate that Complex Graphs can be effective in the comprehension of data semantic. In this Chapter, I will present an approach to automatically generate visual representations of data, coming from a data warehouse, to answer specific questions submitted by a decision maker.

5.1 The proposed approach

First of all, the approach to support the manager in the automatic generation of a Data Warehouse answering to his/her specific requirements will be presented.

For clarity reasons, the following scenario will be considered: the manager (decision maker) expressed one or more questions concerning data contained in several data sources; the manager selected the data sources from internal data sources; at this point, the concepts are extracted in a semi-automatic way starting from the data sources; a data mart has to be generated and presented to the manager based of the complex graph based representation of the data; the manager

Figure 5.1. Data Models associated to the Case Study

takes his decisions.

The selected sources are i) Sales and Marketing and ii) Human Resources data warehouses. The data models associated to these two databases are shown in Figure 6.3.8.

Note that the selected data are imported from the different databases. As so there is the need to perform a data tuning. After the tuning the Apriori Association Rule Mining algorithms implemented in the WEKA toolkit is used to extract association rules. Figure 5.2 shows an excerpt of the conceptual models of the case study, and how their entities are related through relationships representing the minimal set of rules identified by our approach. The Apriori algorithm identified 21 relationships between the attributes of database.

Figure 5.3 shows the generation process, which is articulated in the following phases:

- *Strategic Question Definition.* The manager proposes one or more strategic questions in natural language. It is in charge of the manager verifying if the

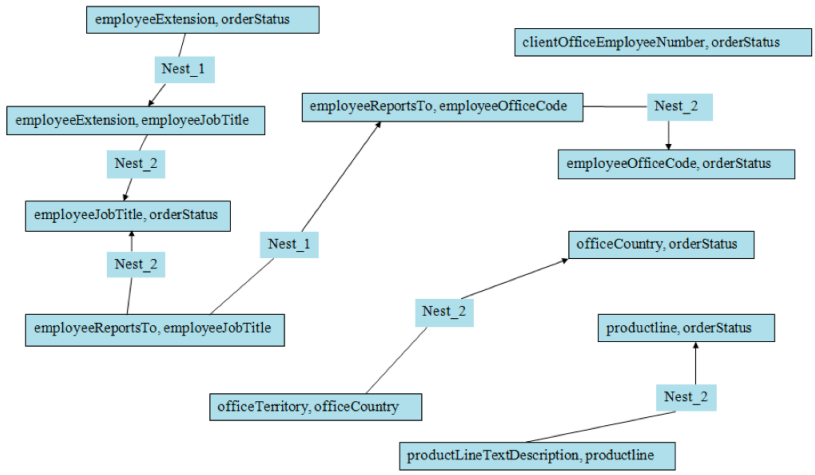


Figure 5.2. CoDe Conceptual Model for the Case Study

entities required to answer the questions are in the available data sources. Indeed, the system provides a description of all the available databases, including internal and external sources.

- *Data Source Selection.* In this phase, the manager selects the heterogeneous data sources that contain the information he needs. The output is the list of databases that will be exploited to create the data wareouse.

- *Concept Identification.* The concepts and their relationships are extracted from all the selected data sources. These concepts are analyzed using an information retrieval approach for selecting the specific concepts (i.e., the entities) that are related to the strategic question.

- *CoDe Modeling.* In this phase, the manager visually selects the concepts (the entities and/or their attributes) represented by information items of the CoDe graphical notation. He can add the relationships among these items taking into account the goal of its data analysis. The results of this phase is

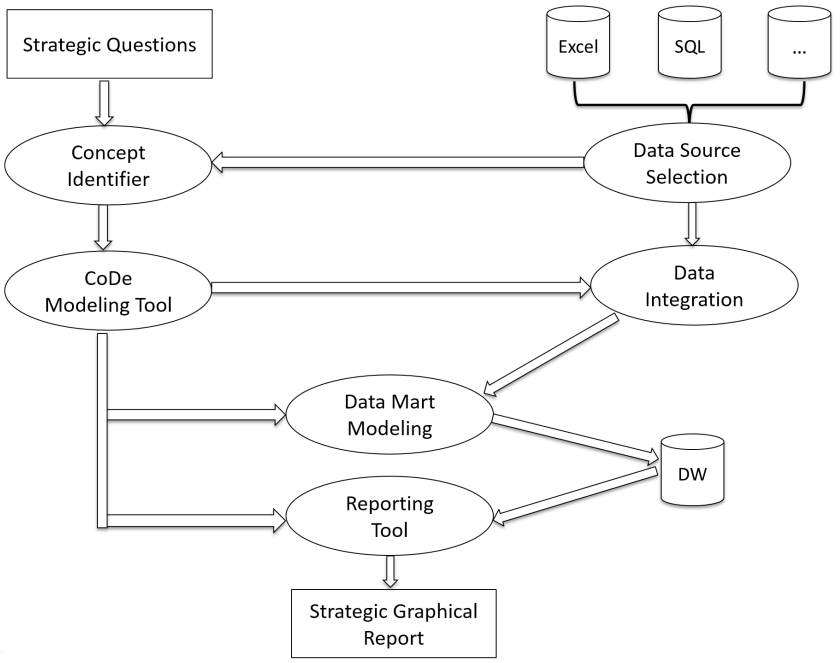


Figure 5.3. The overall process to generate a DW from the strategic questions

a graphical model used to generate the Data Marts in the next step.

- *Data Mart Generation.* The modeling of a Data Mart consists of four substeps. In particular, for each Data Mart:
  - *Data Model Generation*: the Dimensional Fact Model (DFM) is generated starting from the Code Model;
  - *Logical/Physical Model Generation*: a star schema and its physical implementation are generated from the DFM;
  - *Data Integration*: data from the selected data sources are combined and integrates to populate the Data Mart.
- *Report Generation.* Starting from the CoDe model and the DW data, a

graphical report is generated. The report shows all the information the manager needs to make his decision related to the strategic questions.

The single phases of the process are detailed in the following sections through a running example. In particular, we consider an international company that buys collectable model cars, trains, trucks, and buses. The company ships items directly from manufacturers and sells them to distributors worldwide.

5.2 Strategic Question Definition

The manager to direct his business continuously analyzes information about the financial status of his organization, how the organization is currently structured and operating, the level of expertise of his employees, and the customer satisfaction level [46]. Also, external influences have to be understood and taken into account by the decision maker.

A *strategic question* is a question whose answer is needed to perform a strategic decision. A *strategic decision* is a choice which concerns key factors determining the success of an organization's strategy.

Examples of strategic questions related to the proposed case study are:

Q1. Is it necessary to open new offices in the most strategic business areas?

Q2. Is it necessary to hire new employees to improve the customer service?

Q3. In case it is necessary to hire new employees, which are the professional roles the organization needs?

The first question is related to a long term decision that concerns the possibility of expanding the organization in some critical areas. The second question is related to the first. It aims at investigating the need of improving the existing offices hiring new employees. Finally, the third question deepens the second by asking

which kind of professional roles should be considered in case there is the need to add human resources to the company.

All these questions require to extract and relate information about sales trend, sales geographical distribution, organization structure and employees, office performance (e.g. shipping time, selling), employees performance and potential costumer distribution.

5.3 Data Source Selection

In this phase, the manager selects the data sources to be considered to answer the strategic questions. In the case study, the organization has three different internal data sources:

- Sales and Marketing
- Human Resources
- Finance & Accounting

The manager selects the first two. Fig. 5.4 depicts the conceptual models of the selected data sources.

An external data source is also selected: the Collectable Car Consumer Database, provided by an external marketing organization. This database helps collectable model companies in understanding who is buying which models and where. It is based on a market research. In particular, this data source is selected in order to provide enough information to answer question **Q1**.

5.4 Concept Identification

In this phase, the main concepts and their relationships are extracted from different sources. The process uses information retrieval, clustering, data mining and

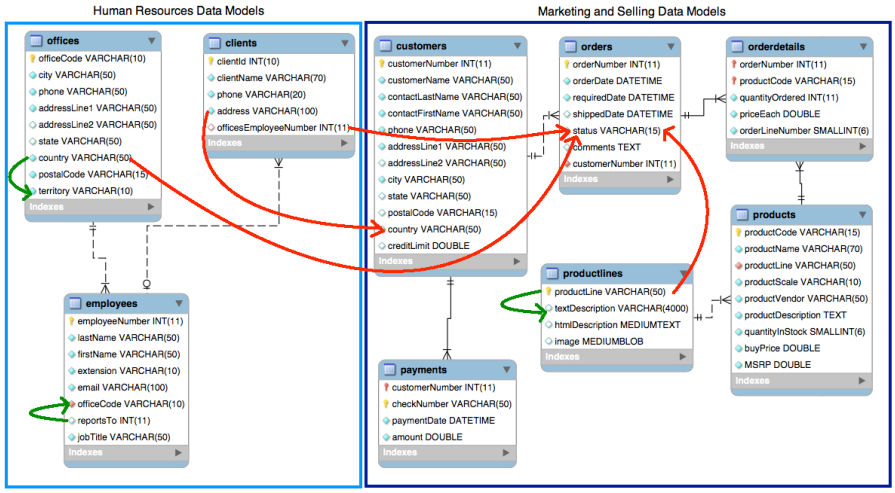


Figure 5.4. An example of extracted rules and relationships identified from the conceptual models

association rule mining techniques.

The concept identification approach presented in this thesis is based on the identification of Approximate Partial Functional Dependencies (APFD) between the subsets of the attributes of a relation. APFDs extend the Type-M Functional Dependencies (MFDs) introduced by [47]. The associative rules we use are briefly described here.

Let $R(A_1, \ldots, A_n)$ a relation. Each attribute A_i has associated a domain, $dom(A_i)$.

Let $\delta : dom(A)^2 \longrightarrow [0,1]$ be a distance function.

Let the *tuple distance function* $g : (dom(A_1), ..., dom(A_n)) \longrightarrow [0,1]$ be such that for $t_1 = (a_1, ..., a_n)$ and $t_2 = (b_1, ...b_n)$ in R

$$g(\delta_1(a_1, b_1), \ldots, \delta_n(a_n, b_n)) = max(\delta_i(a_i, b_i)) \text{ for } i = 1...n$$

where δ_i is a distance function for $i = 1 \ldots n$.

Let R be a relation with attribute set U, and $X \subset U$, with $X = (X_1, ..., X_s)$.

Two tuples $t_1 = (a_1, \ldots, a_n)$ and $t_2 = (b_1, \ldots, b_n)$ in R such that $a_i = b_i$, for each attribute $X_i \in X$, $i = 1 \ldots s$ are said *equal on* X and denoted as $t_1 =_X t_2$.

Definition

Let R be a relation with attribute set U, and $X, Y \subset U,$, with $X = (X_1, ..., X_s)$ and $Y = (Y_1, \ldots, Y_r)$. Let $\tau, \tau' \in$ [0,1]. Let Z the set of the pairs (t_1, t_2) such that $t_1 =_X t_2$ and $m = |Z|$.

$[X] \xrightarrow{t} [Y]_{(g,\tau')}$ is an **APFD - Approximate Partial Functional Dependencies** relation iff for every t_1 and t_2 in Z

$$t_1 [Y] =_{(g)} t_2 [Y] \leq \tau'$$

holds for t% of m, with $t > \tau$, where for $i = 1 \ldots r$. The tuple distance function g is computed as follows:

$$\delta_i = \begin{cases} Levensthein \;\; distance \quad if \quad TYPEdom(A_i) = varchar \\ \qquad (t_1 ! =_Y t_2) \qquad\qquad\qquad otherwise \end{cases}$$

The Levensthein distance between two words is computed as the minimum number of single-character insertions, deletions or substitutions needed to transform the first word into the second one. In particular, the process allows to identify attributes and concepts from data source. The attributes represent a granular module of information and is the same as in the standard ER/dimensional models. A concept represents an entity or a generalization of entities and relationships in the source databases and will be instantiated in the output data warehouse. A concept is formally defined by a name and a finite set of attributes.

The process starts undergoing information to a selection and normalization phase before applying the algorithms on data. Indeed the data mining process is more efficient and easier when data are preprocessed. First, the data is selected and cleaned by eliminating unnecessary attributes. Data preprocessing is used before

the data mining, so that the quality of data is improved and the accuracy and performance of the data mining process are enhanced. In this case the time required by the actual mining is reduced. This operations are in charge of the manager that is the expert of the domain and takes into account the strategic question requirements. Thus, selected data are imported and normalized from different databases, in order to be used by Weka data mining and machine learning software [48]. This operation could be enhanced by applying a data mining technique. In particular, we used classification and cluster analysis data mining techniques. The manager selected the entities *orderdetails*, *customers*, *employees*, *offices* for the question **Q1**.

The classification process divides the dataset into mutually exclusive groups such that the members of each group are as close as possible to each other, and different group members are far from members of other groups. The distance is measured with respect to a specific variable to predict. For example, a typical classification problem is to divide a database of customers into groups that are as homogeneous as possible with respect to a retention variable representing stayer and leaver.

Clustering allows to identify which elements in a data set share common characteristics and group them together based on these characteristics. The most notable difference, however, is that with clustering you let the algorithm determine the groups based on a selection or even all available data, whereas with classification you already have the groups defined. Clusters are computed using a fast, heuristic method that generally produces good solutions. In particular, we applied K-Means clustering algorithm on the transaction data using the Weka toolkit.

The association rule mining finds interesting associations and/or correlation relationship among large set of data items or between two or more entity in a dataset. It infers attribute value conditions that occur frequently together in a given dataset e.g. Market Basket Analysis. Different parameters are used in the

association rule mining for rule generations. In particular, *minimum number of transactions* specifies the minimum number in which a particular item-set must appear so to qualify for inclusion in an association rule. We selected 60 as the minimum support for our work. *Minimum confidence* that basically specifies the minimum confidence threshold for the rule generation. We adopted a minimum confidence value equal to 50. *Lift* is a parameter of interest in the association analysis. Lift is the ratio between the Confidence and the Expected Confidence i.e. the number of transactions that include the consequent divided by the total number of transactions.

5.5 Complex Graphs Generated by CoDe

Once the concepts have been identified and extracted, they need to be represented in an effective way. In this thesis, supported by the findings presented in Chapter 4, the proposed representation is based on complex graphs.

In this section, the basic concepts on the complex graphs are presented and specifed in terms of the syntactic rules offered by the CoDe graphical language. A case study related to the international wholesale distributor of collectable car models is used to provide examples of the notation and the corresponding output.

In many situations managers take decisions considering data related to key performance indicators represented in terms of graphs, combining categorical (nominal) and quantitative data. These data are usually extracted through OLAP

Table 5.1. Output of the Apriori algorithm in Weka

1.	Field3=customerfirstName ==> Field1=customerName conf:(1)
2.	Field1=clientName ==> Field3=customerFirstName conf:(1)
3.	Field1=clientName ==> Field1=customerName conf:(0.79)
...	...

operations from data warehouses and are represented adopting dashboard (standard) graphs, such as bars and pie charts. In particular, a bar chart enables the comparison of the values of each category by examining the lengths of the bars. A pie chart enables the comparison of the proportions of each category to the whole, by observing the angles of the segments. Generally, different aspects of the same problem are represented through separated graphs and does not visualize relationships between information contained in different reports.

Information should be provided to the managers in an efficient way, in such a way to enable them to take quick and effective decisions. According to Bertin [33], one of the most influential theorists in the field of graphic design semiotics,

> The most efficient (graphic) construction are those in which any question, whatever its type and level, can be answered in a single instant of perception, that is, in a single image [33].

A *Complex Graph* follows this guideline. It is a graph connecting standard graphs through graphical relationships. With this approach, conceptual links between data become evident and the interpretability of the data should be improved.

As discussed in Section 5.3, the association rule mining algorithms we use is the Apriori implementation provided by the Weka toolkit. Apriori is used to find frequent patterns, associations, correlations, or causal structures among sets of items or objects in transactions or relational databases. An example of partial output provided by the Apriori algorithm in Weka is shown in Table 5.1.

We use Apriori because of its good performances. In fact, in order to determine the association rule mining algorithm to use, three of the most popular algorithms for the association rules derivation [49] have been compared. In particular, we executed an experiment in which the three algorithms have been executed on two datasets having different attributes, namely the *Supermarket* and *CarShopping* datasets.

First of all, data files have been selected without any missing values. This step is important to avoid meaningless association rules. After data tuning, the Association Rule Mining algorithms is used to extract the association rules. During the comparison, the Apriori, FP-Growth, and Tertius implementation in WEKA have been tested.

After the execution of each algorithm, the number of rules that have been found has been collected. For each algorithm the number of instances, the number of attributes selected, and the execution time is reported.

The results obtained by applying the algorithms Apriori, FP-Growth, and Tertius [50] are summarized in Table 5.2.

It is import to note that, concerning the association rules, performances obtained by the three analyzed algorithms are quite different. In the conducted experiment, the best results have been obtained using the Apriori algorithm.

A comparison of the performance of Apriori and FP-Growth algorithms in generating association rules has been also presented by Hunyadi [51]. In his work, Hunyadi focus on the association rules produced through the two algorithms and found significant correlation between them. Our results confirm the work of Hunyadi. On the other hand, even if the FP-Growth algorithms shows association rules with a positive lift [52], the lift value of Apriori is higher (7 in the example).

Even by comparing the values of Tertius with the results obtained with the Apriori algorithm, we can note that the latter outperforms Tertius for both the

Table 5.2. A comparison between extraction rules algorithms

	Apriori	FP-Growth	Tertius
# Instances	7	7	7
# Attributes	9	53	9
# rules found	10 (visual the best)	73963	300
Execution time	3 sec	5 sec	15 min

positive correlation of the rules and their reliability. Moreover, Tertius is much slower as compared to the Apriori and FP-Growth algorithms.

These algorithms, tend to produce many "useless" rules. Many of them are not interesting, such as the obvious rules, while others may be redundant. For this reason, the whole global association rules can be applied to the algorithm of minimal cover of a set of functional dependencies, obtaining the minimal set of rules [53].

For the purposes of the experiment, it has been used increasingly, the *CarShopping* dataset. As said before, *CarShopping* uses two databases to support its activities, one for the management of human resources (e.g., offices and employers) and one for the marketing and sales management (such as customers, orders or products). This is illustrated in Figure 5.4

In order to make the data mining analysis on the data source of the two schemes, the Cartesian product between every possible pair of tables of the two databases has been calculated. For each Cartesian product, data was loaded into Weka to run the Apriori algorithm, setting the parameters as described before. The association rules relating to the pair of tables involved have been provided as output by the tool. The result is the global set of association rules, making the union of the output of the algorithm Apriori on all Cartesian products. To obtain only the interesting and non-redundant rules, the global rules of the algorithm in Figure 5.2 can be applied obtaining the minimal set of association rules.

Figure 5.1 reports an excerpt of the conceptual models of the case study, and shows how their entities are related through relationships representing the minimal set of rules identified by our approach. The proposed algorithm retrieves the inter-relationships between the same entity and intra-relationships between different entities.

The set of minimal rules can be provided to the manager, who can get a conceptual model of such rules, in accordance with the strategic needs, using the CoDe

visual language.

5.6 CoDe Modeling

The concepts and their relationships identified in the previous phase are exploited to model the Data Mart related to a strategic question. The modeling language adopted to this purpose is CoDe, a visual language based on a logic paradigm [10]. CoDe allows to organize the visualization through the CoDe model which graphically represents relationships between information items and can be considered a conceptual map of the data view.

CoDe is based on the idea that a visual representation of complex knowledge should be considered as a statement of a formal language in the first order logic paradigm [54]. The formal definition of the language specifies both syntactic and semantics rules [24]. Syntactic rules are used to construct well formed visual representations, while semantic rules provide interpretations based on the related information. Thus, a graphical visualization can be considered as made by *terms*, a sort of knowledge items represented by means of suitable graph types, and *relations* between these terms, that point out relationships and state semantics links to the data they represent. According to this paradigm, the CoDe language describes the architectural structure of the visualization at a meta-level, by means of terms and relations, which provide an abstract representation of the information that must be represented.

On the other hand, the same data and relationships can be represented in several different ways, depending essentially on an aesthetic choice of different possible graph types to represent knowledge items, or different styles in rendering the relationships.

However, if the represented knowledge items, and the architectural relationships are the same, by translating these different visualizations in CoDe language, the

equivalent meaning should be obtained as the same expression in this graphic meta-language of abstraction. The aim is to focus on the abstraction of knowledge items, and on the relations between them, provided by the architectural structure of the visualization, and not on the specific shapes selected for the corresponding graphs or relationship links. Indeed, although the shape of an Histogram, Pie, etc., can depend on different aesthetic factors, the meaning carried out is always the same. Thus, in order to define terms, a starting set of basic elements, representing items of information, must be considered. As an example, if an item of knowledge is a distribution of frequencies, it can be suitably represented by a basic standard like an Histogram.

To increase the information carried out by a graph, additional visual elements can be defined, using composition or transformation functions applied to the elements in the starting set. These functions are, in some sense, constructors of graphs that represent more complex information.

Both basic and complex graphs are named terms, and are the elements of the domain of the language. A semantic interpretation of a term is obtained by stating links to the represented data, according to the intended semantics of the graph types.

CoDe considers visualizations that can involve more than one graph. Indeed, by means of suitable functions, graphs can be composed or modified in order to visualize more complex knowledge. In the implementation process of ground data visualization, the starting set of terms is composed of standard-graphs (Histogram, Pie, Area, Bubble, Line, Radar, Bar-Chart, etc.), that can be exploited to visualize a double-entry table. However, any standard-graph has a fixed structural rule that state, by proportional magnitude, the semantics correspondence between the value of a component and its visualization on the plane. Moreover, the final implementation of a standard-graph depends on other visual parameters as: texture, color, orientation, shape, labels, etc.

CoDe also introduces some functions to compose graphs in order to represent more complex information. The visualizations carried out by applying these functions are themselves terms of the language. In general, a function is graphically visualized in CoDe as an oriented arrow connecting the involved terms. A label shows the symbol denoting the function itself.

We consider a report and the associated graph as an information item. Thus, according to the FOL paradigm, we define the representations of these information items as *starting set of terms* of the CoDe visual language.

In the following:

$$T\_name[C_1, \ldots, C_n] \tag{5.1}$$

will denote a report, having T_name as title and $C_1, \ldots, C_n$ as components. The corresponding graphic term of the CoDe language, that visualizes this information item is:

$$\boxed{T\_name[C_1, ..., C_n]}$$

The visualization interface exploits standard graphs (Histogram, Pie, Bubble, Line, etc.) to draw the report. As a consequence, for the same CoDe term, many different visualizations can be given. It is worth to recall that any standard graph has a fixed structural rule that states the semantics correspondence between the value of a component and its visualization. Furthermore, the final shape of a standard graph depends on other visual parameters like dimensions, color, orientation, etc.

Moreover, the visualization interface suitably connects standard graphs representing the input reports, and allows to draw a visualization of the complex CoDe term. In this way, the link with the ground data in the report is stated.

The $NEST\_i$ function has a symmetric definition with respect to the SUM_i function. In fact, it applies to two reports such that the value of one component

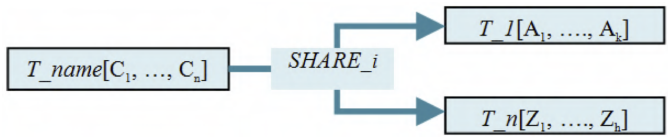
in a report is the sum of the data series in the other report, which can provide a more detailed description with respect to some fixed criterion.

More formally, let T_1 and T_2 be the terms:

$$T\_1[D_1, \ldots, D_h] \tag{5.2}$$

$$T\_2[C_1, \ldots, C_i, \ldots, C_n] \tag{5.3}$$

with $C_i = \underset{j}{AggOp}\, D_j$, where $AggOp$ represents an aggregation operation such as count, avg, sum, max and so on; and D_j their related CoDe terms. A complex CoDe term can be obtained by applying the $NEST\_i$ function from T_2 to T_1. In practice, the $NEST\_i$ function allows to disaggregate a complex component into refined sub-categories. It is important to notie that the arrow has an inverted direction with respect to the SUM_i function.



The CoDe language provides two kinds of relations: *LINK* and *SET*. The $LINK$ relation states the existence of a logical relationship between terms, whereas the SET relation asserts the information given by a single term.

The CoDe terms to denote a link relation between the input terms $T\_1[C_1, \ldots, C_n]$ and $T\_2[D_1, \ldots, D_k]$, is:

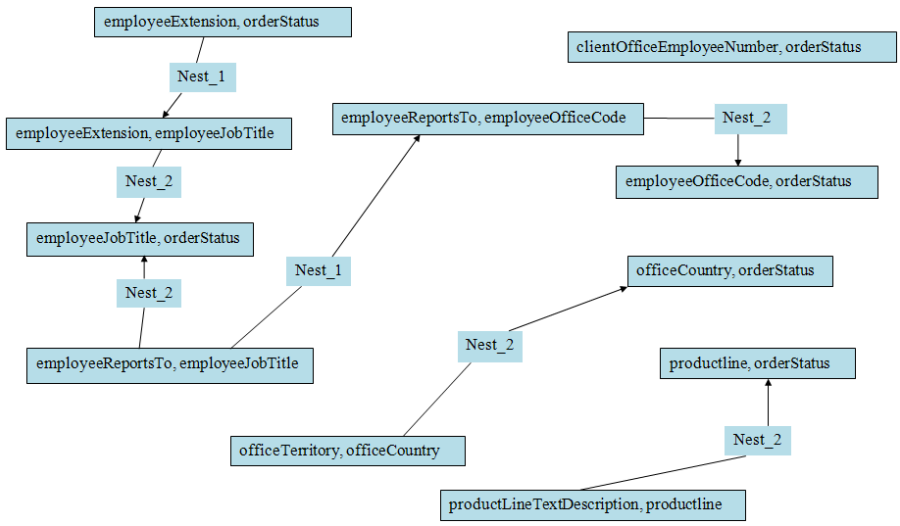


Figure 5.5. The CoDe model

An example of CoDe model is shown in Figure 5.5. The proposed model is not totally automatic, therefore, a subsequent manual work in order to eliminate any redundancies is necessary. This process, requires efforts in terms of data analysis skills as well as of time required for its development.

5.7 Data Mart Generation

In this section the mapping between the CoDe model and the Star schema is introduced. This phase represents the basis of a data mart definition.

5.7.1 Data Model Generation

First of all, it is important to describe how the CoDe conceptual model graph of association rules supports the manager in the identification of multidimensional concepts and construction of the DFM.

From the graphical display made of CoDe and business rules (expressed in natural language), through the automated proposal, the manager can automatically and sequentially enact all the steps necessary for creating OLAP schema for the Data Warehouse Data Mart components.

These phases are: the conceptual design; the logical design; and the generation of the Data Warehouse. The first step consists of creating the OLAP schema of a Data Warehouse, which is the conceptual design phase of the Data Mart that will compose the Data Warehouse. In particular, we deal with the steps of the design and construction of the DFM, for each of the Data Mart, starting from the CoDe conceptual model graph. The entire phase is carried out automatically. The proposed technique for conceptual design of a Data Mart, using the conceptual model of association rules in CoDe according to the DFM, consists of the following steps:

1. Definition of the functional dependency graph;
2. Selection of the facts;
3. For each factor:
   (a) Construction of the related attribute tree;
   (b) Definition of the size / measurements;
   (c) Schema creation.

The functional dependency graph is obtained from the CoDe conceptual model graph. In particular, for each term in the CoDe model $name(C_1, C_2, ..., C_n)$, the following steps must be performed:

1. For each component C_i, such that $1 \leq i \leq n$, create a node in the graph, only if it does not already exist;

2. For all i, such that $1 \leq i \leq n$, create a directed arc from the node associated to C_i to node associated to C_n.

It is worth to note that for each associative rule the right side is composed of only one attribute. In the corresponding CoDe term this attribute will always be represented by the component C_n. Figure 5.6 shows the functional dependency graph obtained from the CoDe conceptual model depicted in figure 5.5.

For the selection of the facts we are running a procedure based on the set of concepts returned from the data in relation to the business rules and the topological characteristics of the functional dependency graph. As a consequence, the result is a selection of facts in accordance with the company's strategic needs.

The procedure consists of the following two steps:

1. Identification of the entities eligible to become fact;

2. Application of a metric for the selection of facts based on the functional dependency graph on the set obtained in the previous step.

Regarding step 1, the set of entities eligible to become fact corresponds to the set of concepts of interest previously obtained. Thus, the set of concepts of interest $E = \{orders, productlines\}$ represents also the set of candidate entities.

Regarding step 2, the metric for the selection of facts is based on the functional dependency graph on the set E. For each entity in E, this step consists of the following activities:

- Select nodes in the dependency graph that represent attributes of that entity;

- Among the nodes obtained in the previous step, select the ones in the dependency graph having only incoming edges;

- If any node in the previous step has been selected, then select only those in the dependency graph having only outgoing edges;

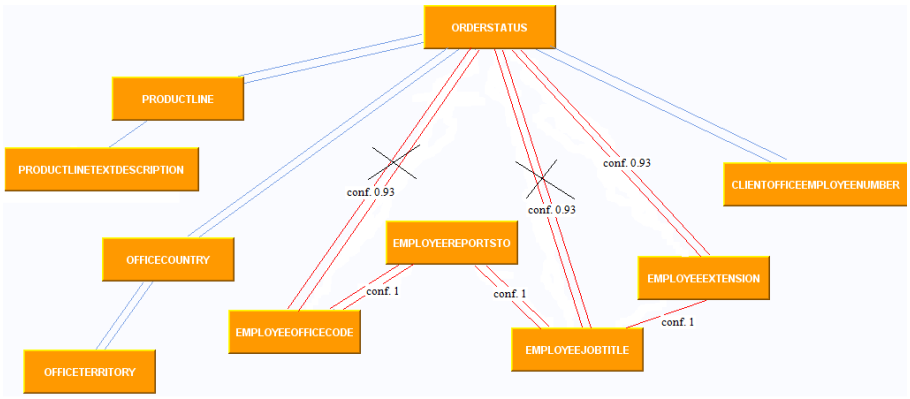


Figure 5.6. Tree attributes for the fact *orders* and elimination of cycles

- In case more than one node has been selected, arbitrarily choose one of them;
- The chosen node represents the fact.

By performing this procedure for the current case, the following nodes are selected: orderStatus, productLineTextDescription, orders and productLines.

A tree of attributes for each of the selected facts is built from the functional dependency graph. Given a node F, designated as the representative of a fact, we define the attribute tree as follows:

- Each node in the functional dependency graph corresponds to a node in the tree;
- For each directed edge (u, v) or (v, u) between two nodes u and v in the dependency graph, we define an undirected link between the corresponding nodes u and v in the tree. This arc is single if the cardinality of the association between u and v, the source schema, is of the type *: 1, otherwise if the cardinality of the type * N is a multiple of the link.

The so constructed tree attributes may contain pairs of nodes that can be connected by two or more distinct paths, giving rise to a sort of loop. A tree is

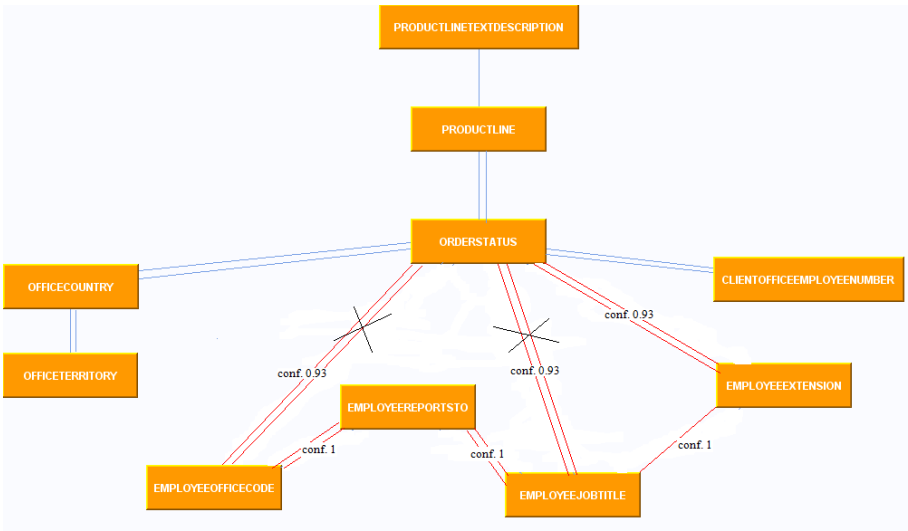


Figure 5.7. Tree attributes for the fact *productlines* and elimination of cycles

defined as an undirected graph in which any two nodes are connected by one and only one path (i.e., connected undirected graph with no cycles). So all cycles must be removed. The proposed procedure, recursively iterates as long as there is at least one cycle in the graph. For each iteration, one of the link involved in the cycle is removed. We choose the link representing a membership rule with the lowest value of confidence. In this way, we keep those links with rules that have a greater degree of certainty. Once all cycles are removed, the procedure ends. Example of resulting trees are shown in Figures 5.6 and 5.7, in which links forming cycles [1] are highlighted in red while links removed from our cycle removal procedure are marked.

The selected dimensions determine how events are aggregated to the decision-making process. They should be selected in the tree of attributes among root

[1]For each of them the value of the confidence of the rule represented associative is indicated.

node children. They may correspond to discrete attributes or interval (discrete or continuous).

In the example, by observing the tree in Figure 5.6, when considering the fact

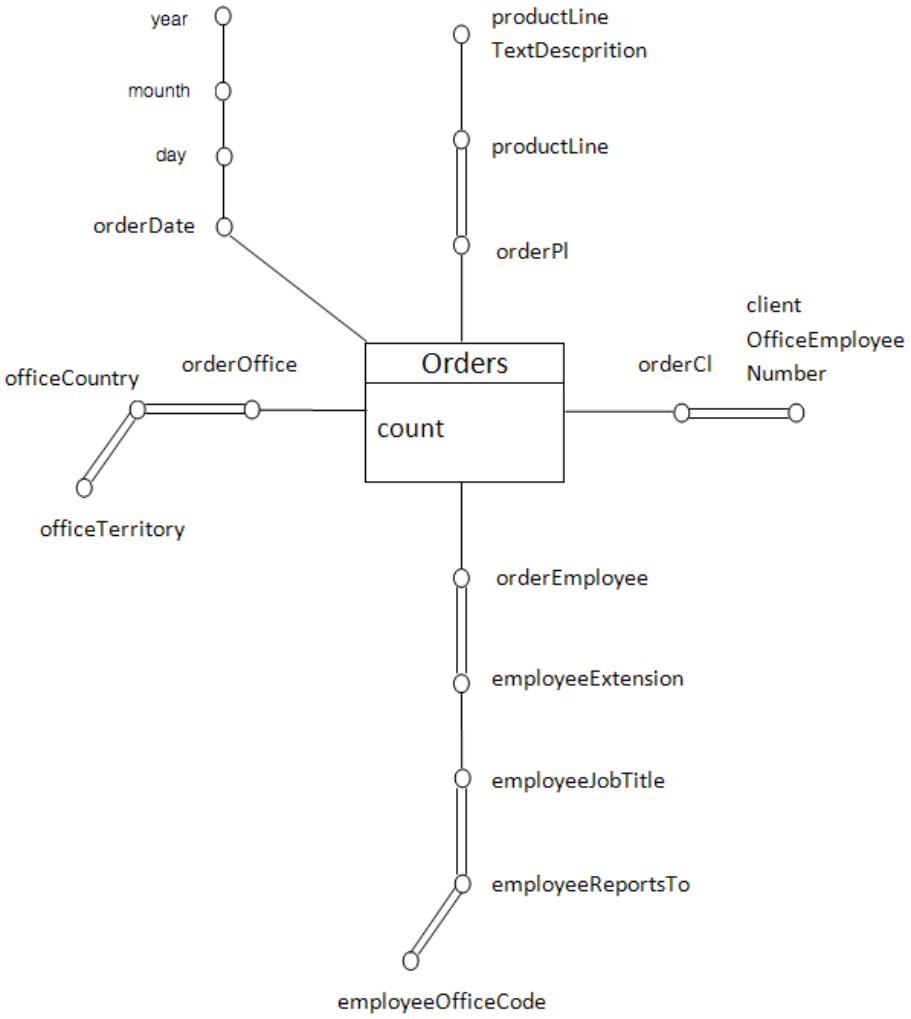


Figure 5.8. Scheme for orders

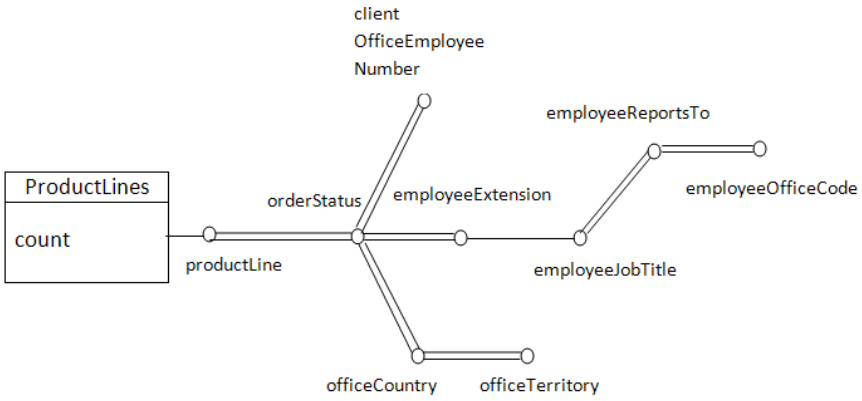


Figure 5.9. Fact scheme for product lines

orders, the dimensions are: officeCountry, employeeExtension, clientOfficeEmployeeNumber, and ProductLine. Observing Figure 5.7, for the fact *productLines* the dimention is ProductLine.

A measure is a numerical property of a fact and describes a quantitative point of interest for the analysis. Therefore, measures should correspond to numerical attributes. We can apply aggregation functions to sum, average, maximum, minimum.

In our case, for the fact *Orders* and *productLines*, we do not have numerical attributes to combine. So, we define for these facts a predefined measure *count*, that performs the count aggregation operator.

We can aggregate data along the time dimension. If the schema does not actually define a temporal dimension, it is possible to add it by selecting all the date fields in the schema. It is allowed to the user choosing among the dates to add. In our example, there is only one date field, i.e. the attribute *orderDate* of the table related to the entity represented by the fact, to which the size of *Orders* (orderDate) is added manually to the schema. In figure 5.8 the schema for the fact *Orders* is shown, while Figure 5.9 shows the schema for the fact *productLines*.

5.7.2 Logical/Physical Model Generation

Once the scheme have been defined, for each of the identified facts the conceptual design phase of the Data Warehouse can be considered complete. It is worth noting that the main steps of this phase can be all carried out sequentially and automatically. This enables the first step of automatic generation of the OLAP schema for the Data Warehouse.

5.7.3 Data integration

Generally speaking, the word "integration" is frequently used to denote a process that forms the whole out of a multiple parts. The term "data integration" often denotes the process that combines data from different sources to provide a single comprehensible view on all of the combined data.

A typical example of data integration would be combining the data from the inventory system with data from the sales to allow fulfillment order to be directly related to changes in the inventory. Another example of data integration is merging customers and contacts data from separate departmental CRMS[2] into a corporate-wide system.

One way to understand data integration is to decompose the process of filling the data warehouse into a number of distinct activities. To perform the Data Integration we use a data integration engine. This engine is a software component that is capable of interpreting and executing jobs and transformations, thereby performing the actual data integration tasks accordingly.

A *transformation* consists of a collection of steps. A *step* denotes a particular

[2]The Customer Relationship Management(CRM) is a business strategy used by several companies to identify and manage profiles and prospects of their customers, so as to develop activities and strategies that, on the one hand, help to capture new customers and maximize profits and, on the other hand, create a customers loyalty, trying to understand their needs and expectations. Such a strategy is usually supported by CRM systems (CRMS).

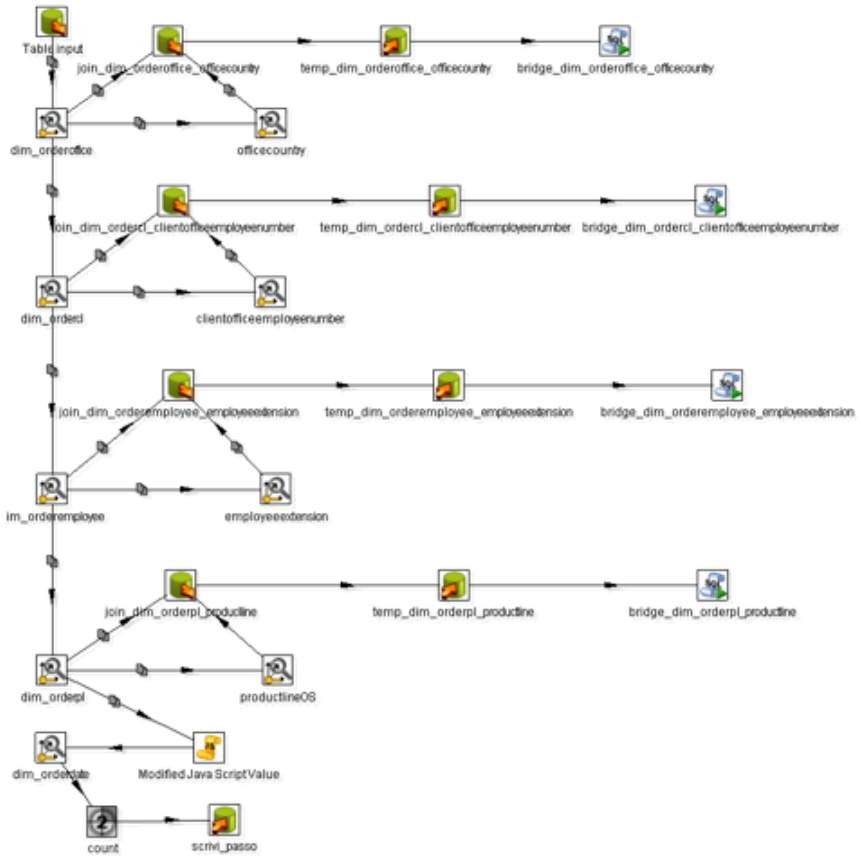


Figure 5.10. The data integration transformation for the fact *Orders*

operation on one or more record streams. Steps may be connected by hops. A *hop* is like a pipeline through which records can flow from one step toward another step. A *record stream* is a series of records. A *record* is a collection of values that is structured in such a way that each value is associated with exactly one field. The collection of fields associated with all values in a record is called the *record type*. All records in a record stream must be of the same record type.

Typically, jobs are composed of one or more transformations (see Figure 5.10 for an example of data integration transformation). For example, to load a star

schema, you would typically build one transformation to do the actual extraction, and build one transformation for each dimension table, and one transformation to load the fact table. The job would be used to put all these transformations in the proper sequence (first extract, then load all dimension tables, and then load the fact table). Like transformations, jobs consist of a number of interconnected items, but the likeness ends there. Figure 5.11 shown an example of data integration engine, namely Pentaho[55].

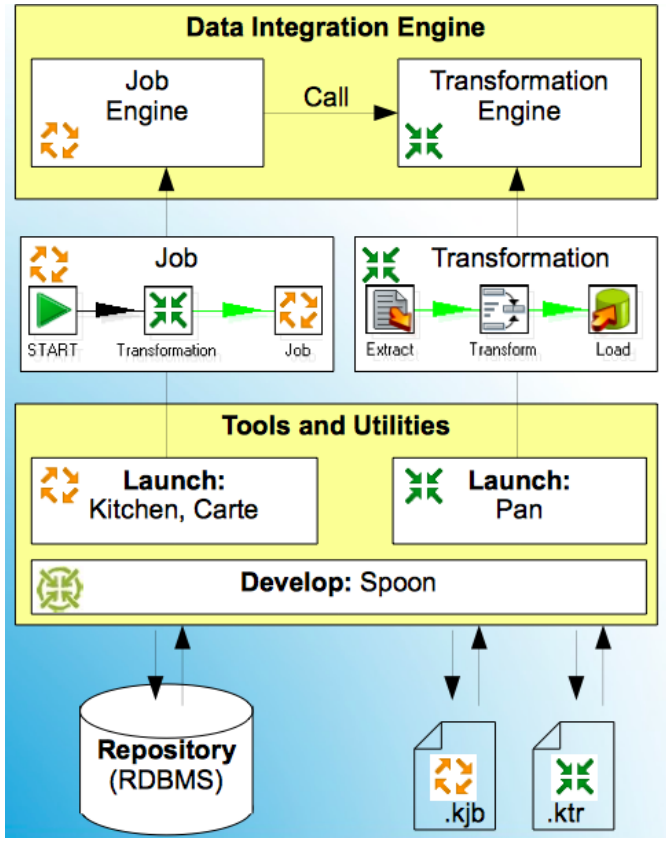


Figure 5.11. Data Integration Engine example

5.8 Reporting Tool

Once the OLAP scheme for the Data Warehouse has been realized, the CoDe model can be used as a Reporting Tool to generate a strategic graphical report containing the information about strategic needs that support the manager during the decisional process. The reporting activity is supported by the manager that selects the Data Warehouse after he obtained concepts and relations considered useful to answer his needs. Thus, the proposed methodology can be divided into two main phases:

- extraction, integration, and modeling of the relevant information about the strategic needs from the data sources;
- automatic generation of the OLAP schemata from the previous step information.

A CoDe term representing an association rule $Field \rightarrow Field1$ can be considered as a double entry table with two columns *Field* and *Field1*. Such a table can be visualized using standard diagram (histograms, pie charts, etc.). Consequently, we can verify if the obtained cube satisfies the strategic needs by representing any item of the model in Figure 5.5 with a standard diagram.

To this aim, for each item we execute an MDX query[3] that selects the attributes contained in the item. The results are used to generate a report by means of vertical bars. In Figure 5.12 the model representing any item of the CoDe model is represented with vertical bar graphics. This reproduces the model of Figure 5.5.

[3]Multidimensional Expressions (MDX) is a query language for multidimensional OLAP databases. MDX queries act on multidimensional objects, such as cubes, and return multidimensional cell sets that contain the cube's data.

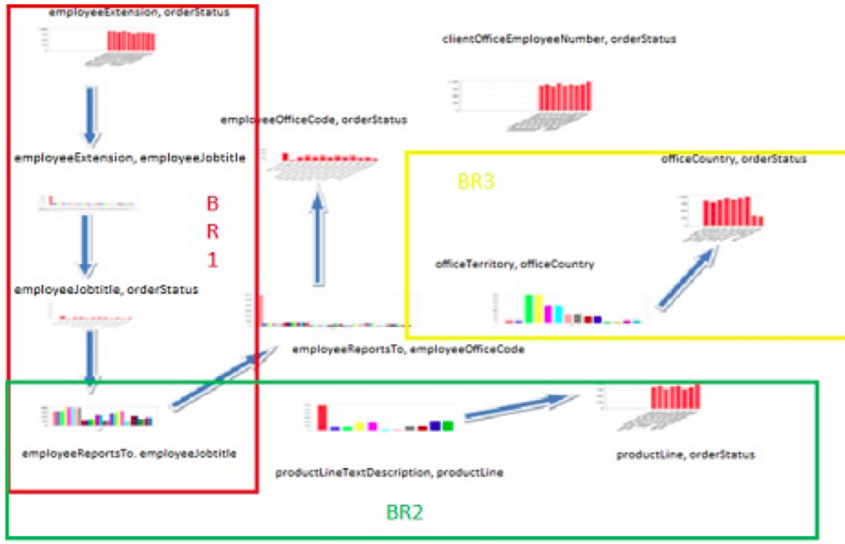


Figure 5.12. Graphical representation of the CoDe model generated

Chapter 6

Semi Automatic Data Integration

During the assessment of the proposed process, the most critical phase of the process was the data integration. That phase, in fact, needs of a major contribution from the manager. At this point, to encourage the implementation of the process in a real setting, it is important to provide support to the manager during this phase. Unfortunately, the data integration phase cannot be completely automated, but a semi-automatic process is certainly possible and valuable.

In this Chapter the semi-automatic data integration process will be presented together with some of the case studies we used to assess the new approach.

6.1 Semiautomatic Data Integration Process

The study of the review of literature has revealed that most of the related research does not specifically address the integration of heterogeneous data sources, and most of the authors [56] [57] [58] [59] [60] does not provide enough insight to understand how they derive the integrated schema. To the best of my knowledge, no method is available to automatically populate the schema using integrated data.

On the other hand, there is no incentive to integrate data and OLAP techniques to construct a single, ready-made and targeted schema to meet the strategic requirements imposed by the manager in a subjective manner.

In this chapter I will present a methodology that defines a semi-automatic process, to support business managers during this phase. Starting from the different sources of related data, the approach enables their integration in a single reconciled data source that can be used to build the OLAP schema.

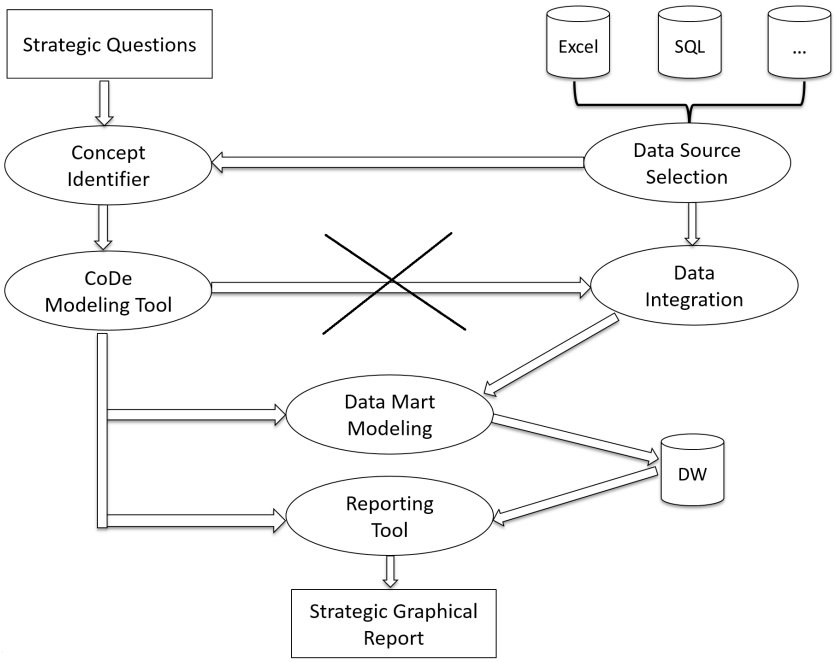


Figure 6.1. Generation Process

During the process of source integration as well as the construction of the OLAP schema, the end users are not aware of the process backend. They are only asked to provide an acceptable error thresholds for the final integration. As a consequence, users do not have specific knowledge of datai integration or data warehousing.

In order to validate the integration process, they will obtain an OLAP ready-to-analyze scheme and set the error margin. Indeed, the process was designed to build OLAP schemas starting from the integration of related data sources and modeling any domain of the real world without focusing on one in particular. Basically, the success of the process is related to the fact that data sources are related to each other. Is this is not the casa, of course, the integration is meaningless. The fact that user has to establishing an error thresholds greater than zero, is due to the nature of the techniques used in the process itself, that will be presented in the next sections.

6.2 Integration process: from data sources to OLAP schema

The methodology of the process is depicted in Figure 6.2 shows . Note that the process can be split in two main sections or steps:

1. Extraction, integration of information from data sources, and building the reconciled scheme;
2. OLAP schema generation, starting with the reconciled source pattern obtained in the previous step.

The flowchart diagram in Figure 6.2 show the two main steps of the process using different colors. Note that the user interaction is limited to the configuration (i.e., the selection of the error threshold) and the specification of the strategic questions during the successive phases of the process.

The next paragraphs describe in detail each step fo the process. To better clarify the proposed process, I will assume a sample case study composed of only two correlated data sources belonging to the same domain. However, the process can be generalized to more complex cases involving an higher number of sources.

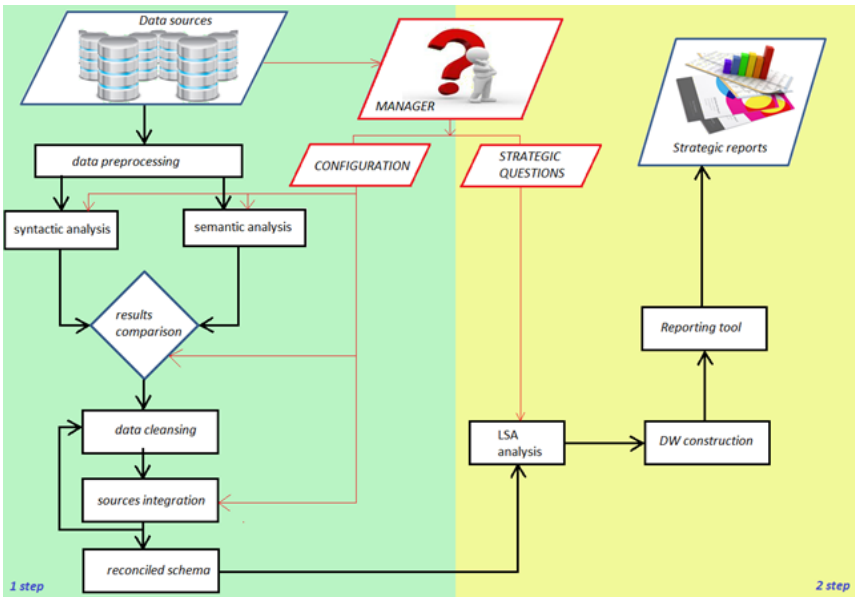


Figure 6.2. Flowchart of the data integration process

6.3 Data extraction and integration and construction of reconciled scheme

This section describes how to extract information from data sources through Data Mining and Machine Learning techniques, how data is integrated and used to produce a new reconciled schema.

6.3.1 Data Sources

The aim of the first step is to build an integrated schema of heterogeneous data sources under appropriate conditions. A first condition required for the success of the outlined process is that data sources are correlated, i.e., they model the same domain. This condition is of utmost importance because intuitively the integration of spatial databases on different domains does not make sense [59]. However, if the

different data sources model independent and distinct portions of the real world, the problem of integration would not exist. Thus, the condition probably holds in most real examples.

In the illustrating example, let us consider an international company, named *Car Shopping*, that buys vehicles (car, train, truck, bus) and sells and ships them from manufacturers to distributors all over the world. *Car Shopping* uses two relational databases to support its business, the first containing human resource management (e.g., offices and employers) and the othe for marketing and sales (e.g., customers, orders, and products). Let us refer to the databases as $HUMAN\_RESOURCES$ and $MARKETING\_AND\_SELLING$, respectively. See Figure 5.4 for a starting data model of the example.

Records contained in the data sources are extract (e.g., tables from relational databases are exported) and are stored in CSV files (Comma Separated Values). Each CSV file will be named as the database table name. As an example, the client table in the $HUMAN\_RESOURCES$ database will be stored in the *clients.csv* file.

6.3.2 Data preprocessing

The second transformation to be applied to the data concerns the representation of text documents into an algebraic model so that they can be further processed. This is necessary to enable scheme matching by highlighting common concepts between data sources. This is a challenging task for many reasons[59]. Firstly, schema modeling the same concept (entity, relationships, or attributes) may be different in the structure and names chosen by the designer (e.g., synonymy). Secondly, similar words can have different meanings (e.g., polysemy). As a result, it is necessary to preprocess the data before reaching the possible matching between the schemes.

In this thesis, the documents refer to to the entities, and then to the tables of the relational databases, i.e., the data sources to be integrated. These documents

form the input of the text mining process, consisting of a textual analysis to extract information from input data. The input documents constitute the *corpus*, i.e., the collection of documents to process. After obtaining the corpus, text needs to be normalized to improve the performance of the successive phases. In particular, the following operations are applied:

1. removing columns of tables having many null values (sparse fields);
2. punctuation and white space removal;
3. converting each word into lowercase;
4. execution of stop-words (e.g., articles and prepositions) and stemming algorithms to remove noise and normalize words.

If a column of a document has many null values, it can be removed because it does not provide enough information to highlight similarity between documents. Moreover, the greater the final matrix the higher the time complexity.

Punctuation and white space removal operations are useful for deleting everything that does not give information to the text. Each word is then converted to lower case by convention. Words that do not contribute to corpus information can increase the noise, causing worse IR performances. Passwords, for instance, can be eliminated using specific algorithms, since they do not provide useful information to discriminate the topics discussed in a document. The stop-word expression was coined by Hans Peter Luhn, an Information Retrieval pioneer, to refer to specific words (contained in a stop-words list) that can be removed from a document without affecting the semantic of the text. On average, removing stop-word reduces the size of the corpus of $20 - 30\%$.

The stemming is the process of reducing the use of a word (e.g., its conjugation) to its root form, called the theme or stem. The theme does not necessarily correspond to the morphological root (the lemma) of the word, however, for our purpose

it is just important that related words are mapped onto the same theme. For example, the words "fishes", "fishing", "fished", "fishy", and "fisher" are all stemmed to the root word "fish". Similarly to stop-word, stemming reduces the size of the corpus.

	D_1	D_2	⋯	D_j	⋯	D_n
T_1	occ(1,1)	occ(1,2)	⋯	occ(1,j)	⋯	occ(1,n)
T_2	occ(2,1)	occ(2,2)	⋯	occ(2,j)	⋯	occ(2,n)
⋮	⋮	⋮	⋯	⋮	⋮	⋮
T_i	occ(i,1)	occ(i,2)	⋯	occ(i,j)	⋯	occ(i,n)
⋮	⋮	⋮	⋮	⋮	⋮	⋮
T_m	occ(m,1)	occ(m,2)	⋯	occ(m,j)	⋯	occ(m,n)

Figure 6.3. m x n term-document matrix

After the preceding steps have been enacted, we can build the terms-documents matrix, that is, the matrix that provides the frequency distribution of each term within the various documents. This matrix serves as a basis for many complex statistical analyses. In algebraic terms, given n documents $D_1, D_2, ...D_n$ to analyze (the corpus) and m terms $T_1, T_2, ...T_m$, the cell of the terms-documents matrix tdm [i, j] reports the number of occurrences of the term i in document j, i.e., occ (i, j). Figure 6.3 shows a sample term-document matrix.

The term-document matrix is represents the first step towards a mathematical representation of the corpus. The occurrences of terms in the documents are usually weighed before being further elaborated. The overall weight assigned to each entry of the matrix depends on two components: the frequency weight and the term weight. The frequency weight is a value that depends on the frequency of the individual term, or *term frequency* (tf), while the term weight that takes into account the number of times the word appears in the document collection, or *document frequency* (df). If we are interested in infrequently occurring terms, we

can calculate the inverse of the document frequency (idf). In particular, the term frequency-inverse document frequency (tf-idf) value is used in Information Retrieval to measure the importance of a term compared to a document or collection of documents. This value increases with the number of times the term is contained in the document, but is inversely proportional to the frequency of term in the collection. The underlying idea behind this behavior is to give more importance to the terms that appear in the document, but that do not appear in other documents. The function can be split into two factors: the first factor of the function is the number of terms in the document. This value is divided by the length of the document itself to avoid that longer documents are privileged. In algebraic terms, the tf-idf function is given by factors:

$$tf_{i,j} = \frac{n_{i,j}}{|d_j|}$$

where $n_{i,j}$ is the number of occurrences of the term t_i in the d_j document while the denominator is simply the dimension, i.e., the number of terms, of the d_j document. The other factor of the function indicates the general importance of the term in the collection:

$$idf_i = \log \frac{|D|}{|\{d\, t_i \in d\}|}$$

where $|D|$ is the number of documents in the collection, and the denominator represents the number of documents that contain the term t_i. Thus, the measure is:

$$(tf - idf)_{i,j} = tf_{i,j} \cdot idf_i$$

It is important to note that the construction of the term-document matrix does not take into account numeric data. The term-document matrix is the input for the two successive and parallel phases, namely syntactic and semantic analysis.

6.3.2.1 Preprocessing Data in R

The whole process was implemented in the R language using the CRAN (Comprehensive R Archive Network) libraries for text mining (e.g., the tm library of the Text Mining package). Figure 6.4 4 shows a snippet of the code used to build the terms-documents matrix.

```
library(tm)
library(stringdist)#per la levenshtein
ds<-DirSource("C:\\Users\\Angelo\\Desktop\\client&customer\\file_csv")
#An object of class Corpus contains collection of text documents
corpus<-Corpus(ds)
#elimina gli spazi bianchi
corpus<-tm_map(corpus,stripWhitespace,mc.cores=1)

# remove punctuation
corpus <- tm_map(corpus, removePunctuation,mc.cores=1)

#Eliminating Numbers
corpus <- tm_map(corpus, removeNumbers,mc.cores=1)

# make each letter lowercase
corpus <- tm_map(corpus, tolower,mc.cores=1)

# remove generic and custom stopwords
#corpus <- tm_map(corpus,removeWords, c(stopwords("english"),'\t'))

#corpus<-Corpus(VectorSource(wordStem(corpus,"english")))#sotto linux per lo stemming
library(Snowballc)
 #corpus<-tm_map(corpus, stemDocument)
#term document matrix
#applicare le misure TF-IDF con  weighting = weightTfIdf
tdm<-TermDocumentMatrix(corpus, control = list(removePunctuation=TRUE,
                                               removeNumbers=TRUE,stopwords=TRUE,encoding="UTF-8"))
```

Figure 6.4. Snippet of the code to build the terms-documents matrix

In particular, the Corpus function allows the construction of the corpus of the current documents in the directory, the *tm-map(x, f)* function applies the f function to each document in the corpus x Here, f is a preprocessing function text (removePunctuation, removeNumbers, etc.). The *TermDocumentMatrix (x, control = list ())* function builds the term-document matrix x for the corpus. It is possible to tune some options, such as the weights, to manage *tf-idf*. By default weighting is tf (term frequency).

The input of the procedure is composed of CSV files containing the source $HUMAN_RESOURCESand$MARKETING_AND_SELLING tables schemata, as described before. In case of homonymous tables the filename is composed of the union of the database name and the table name.

In the example, the term-document matrix is composed of 1361 x 8 entries (terms x documents). The $MARKETING\_AND\_SELLING$ order details is not considered because containing only numeric data.

6.3.3 Syntactic analysis

A first possible data analysis is the syntactic analysis on names, data type, domain and descriptive information. The first studies on the matching schemes topic, were mainly based on this kind of information [59]. In our process, the input of the syntactic analysis is the term-document matrix. The syntactic analysis is divided into:

- Computation of the Levenshtein distance,
- Cluster analysis.

6.3.3.1 Computation of the Levenshtein distance

The Levenshtein distance is computed between all corpus documents to find syntactic similarities. Precisely, each column of the term-document matrix is considered separately. Thus, each column (representing a document) is a vector and we calculate the Levenshtein distance among all the possible pairs of vectors (documents). The output of this phase is a matrix of the distances.

6.3.3.2 Cluster analysis

The Levenshtein distances matrix is the input for the cluster analysis phase, whose goal is to understand how documents are deployed in the various clusters due to their syntactic similarity. Two types of clustering can be considered:

- Hierarchical clustering
- Non-hierarchical clustering

Hierarchical clustering can be used to display the dendrogram, a simple global vision of the possible partitions of the documents. However, since we want to automatize this phase, we do not use this approach because it requires the manager to decide when the hierarchical analysis should be stopped, i.e., at which level to cut the dendrogram. An example of dendrogram produced by a hierarchical clustering is shown in Figure 6.5.

Instead, the k-means algorithm is used by setting a priori appropriate parameters, namely:

- Number of clusters (k) equal to the half of the number of documents in the corpus;
- Number of iterations equal to 10000;

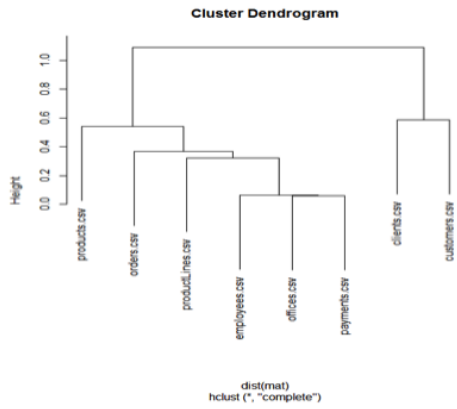


Figure 6.5. An example cluster dendrogram

- The Levenshtein distance matrix obtained in the previous phase.

To facilitate the analysis of the huge amount of data and to ensure the stability of the algorithm itself, we define the following constraints: (i) the k-means algorithm ends when there is no improvement in the index $between\_SS\ /total\_SS$ in five consecutive iterations; (ii) for each execution of the algorithm, delete the document from the resulting cluster; the new set of documents, diminished by one unit, will be the input for the next iteration of the algorithm and the threshold will be decremented by one; (iv) the minimum number of possible clusters is 2.

Choosing these constraints significantly affects the outcome of data Integration. This constraints have been derived from heuristics available in the literature and various tests we conducted. The output of this algorithm will be crucial for the remaining phases of the process.

6.3.3.3 Advantages and disadvantages

Syntactic analysis is simple to implement, but its precision is not optimal, so there could be false positives. Syntactic computation is able to find candidate documents to be included, but there may be other documents dealing with the same concept having a different syntactic form (for example, abbreviations, synonyms, different languages, etc.). To cope with these problems, it is necessary to introduce an analysis that takes into account the context of the documents to be integrated, thus implementing a semantic analysis.

On the other hand, the advantage of using this approach is the speed of execution that allows a quick first examination of the data. For these reasons, the process uses a combination of the two types of analysis.

6.3.3.4 Syntactic analysis implementaion

To calculates the Levenshtein distance between two pairs of input vectors A and B, we can use the **stringdist(A,B,method ="lv")** function implemented in R. The input vectors are the columns vector of the term-document matrix, each representing a document. The element of the resulting distance matrix $M[i, j]$ will contain the average Levenshtein distance calculated on each pair of elements composed of the respective vectors i and j.

Figure 6.5 shows the dendrogram obtained with the hierarchical clustering for the case study. It can be noted the existing hierarchy between the documents, highlighting syntactic similarities between them. For example, a syntactic similarity can be supposed between clients and customers tables of the $HUMAN_RESOURCESand$MARKETING_AND_SELLING databases because they are inserted in the same cluster. Similarly, the cluster formed by employee, offices, and payments suggest a syntactic similarity among their schemata.

On the other hand, *kmeans* algorithm in R is available by installing the stat package. Such implementation returns a value that indicates fitting goodness for input data, $between\_SS/total\_SS$. In particular, $total\_SS$ is the sum of the squares of the distances of each data point from the global sample average (or centroid); $between\_SS$ is the sum of squares of distances between centered groups obtained from the global sample size, seen as the "center of gravity" of points (in this calculation, multiply the square of the distance of each centroid from the global average for the number of data points in the cluster).

A high value of $between\_SS/total\_SS$ means that clusters are well separated and the inter-cluster variance is low: This index has values in the range [0-1] where 0 indicates the perfect overlap of the clusters, and hence a result to be discarded, and 1 indicates a good clustering.

According to the constrains mentioned before, in the considered case study the

Table 6.1. Corpus documents fro the example case study

Document	ID
clients	1
customers	2
employees	3
offces	4
orders	5
payments	6
productlines	7
products	8

Table 6.2. k-means execution for syntactic analysis

Iteration	Parameters		Documents ID	Obtained clusters	between_SS / total_SS (in %)	Document to delete
	k	n				
1	4	8	1,2,3,4,5,6,7,8	C1={1,2} C2={3,4,6} C3={5,7} C4={8}	86.8	8
2	3	7	1,2,3,4,5,6,7,8	C1={1,2} C2={3,4,6} C3={5,7}	83.4	-

k-means algorithm executed on the Levenshtein distance matrix obtained during the previous phase, used an initial threshold k=4 (in the middle of number of documents in the corpus) and maximum number of iterations equal to 10000 for each k-means execution. For convenience, the corpus documents are numbered in Table 6.1. Table 6.2 summarizes the execution of the algorithm for the case study. Documents are referenced by the number assigned to them. Note that two executions of the algorithm are needed. The algorithm clustered the first seven documents in three clusters, with a fitting goodness of about 83%. The clusters C1, and C2 obtained during the second iteration suggest the existence of a matching between the two source schemas.

6.3.4 Semantic analysis

During this phase, the output of the previous step are analyzed to find semantic similarities. In particular, an unsupervised classification and clustering is performed in the data. Unsupervised learning is one of the main data mining techniques, that starting from an input (the system experience) reclassifies and organizes it based on a set of common features to make reasoning and predictions on subsequent inputs. In our case, the input consists of the previously preprocessed database tables (i.e., the term-document matrix), while the learning outcome is a Self-Organizing Map (SOM) neural network, a special unsupervised learning neural network that produces a topological cluster-mapping on a plane (two-dimensional) or in space (three-dimensional) staring from the training data. we want to discover the existing semantic correlations between the tables of two (or more) databases, even in case of different technological nature (relational vs reticulated) and/or naming conventions (eg, client vs. customer), but modeling the same concept of the application domain.

Semantic analysis is divided into phases:

1. SOM network training;
2. Calculation of the weight covariance matrix assigned by the SOM to the documents;
3. Clustering analysis.

SOM Network Training. The reduced term-document matrix is the input, also called the training set for the SOM network. The output of this phase is a two-dimensional lattice, proportionate to the number of documents (tables) analyzed. Each cell, or SOM unit, contain similar documents, as well as close cells contain groups of documents that share some feature. The resulting lattice can be plotted to show the input allocated to SOM units on a two-dimensional plan.

Calculation of the covariance matrix weights assigned to the documents by the SOM.

Semantic analysis continues by performing an unattended classification of the output of the SOM network, that is, of the output weighing matrix assigned to the documents. Concretely, the output weights calculated by the SOM form an (n^2 x n) cardinality matrix, where n is the number of documents processed by the SOM.

This matrix is submitted to a normalization step so that weight values are real numbers within range [0,1]. From the normal matrix are also excluded those elements that do not exceed a threshold set by the user of the Information System, as it means that the excluded element contribute to the SOM unit is marginal.

The normalized and reduced matrix is split by columns (i.e., the documents). Each column vector will have n^2 components (the i^{th} component represents the weight assigned by the SOM to that vector for the i^{th} unit in the lattice). Than, covariance is calculated for each pair of column vectors obtained. The covariance matrix is quadratic in the number of documents, indeed each pair of possible documents is connected to the other, representing the variation of each variable compared to the others.

The covariance matrix obtained will be the input for the cluster analysis phase.

Clustering analysis.

The last phase of the semantic analysis is the clustering, whose goal is to understand how documents are separated in the various clusters by semantic similarity given the covariance matrix obtained at the previous stage. This phase is similar to the one performed in syntactic analysis, the same methods and constraints are used.

6.3.4.1 Advantages and disadvantages

Semantic analysis takes into account the context of data usage. Moreover, unsupervised learning from data, where structural and technological aspects of the sources

are unknown, may highlight concepts hidden in the data themselves. In addition, the advantage of using this technique as well the two-dimensional projection of the lattice, and hence the graphic representation of the identified clusters, hides the complex learning phase from the data and the definition of similarity between them to the user.

Overlooking the technical aspects of the SOM, which will be shown below, the crucial aspects that encourage the use of such technology are: the ability to convey a large amount of information in a limited space; the easy of navigation and perceptual inferences on exposed recovery interfaces; and the possibility of finding semantic relationships between the data [8].

Disadvantages of using SOM exist. In fact, the existence of a network that solves the problem in not guaranteed, because there is not always a learning algorithm that converges by giving a low error network output. Moreover, output values could not be accurate, but they have range in which they can vary. A very large case study is needed to obtain good learning and low output error. Furthermore, the reason why a specific result is given is unknown. Finally, sophisticated training techniques require a lot of computing time, thus requiring high performance machines.

6.3.4.2 Semantic analysis implementation

An implementation of supervised and unsupervised SOM networks in R is available using the Kohonen package. We used an unsupervised SOM as we want to automatize this phase as much as possible. The setting of some thresholds are the only input requested to the manager. Figure 6.6 shows the code to build the network.

The *som* function takes in input the normalized matrix of covariances in which the terms have been removed and builds a lattice quadratic in the number of input documents (the number of columns in the covariance matrix is equal to the number of columns in the term-document matrix). In our case we have 8 documents (see Figure 6.6) so the pattern will consist of 64 SOM units. The

```
...|
library("kohonen")
set.seed(7)
psom <- som(rider, grid = somgrid(8, 8, "hexagonal"),keep.data = TRUE)
#visualizzazione grafica della rete som
coolBlueHotRed <- function(n, alpha = 1){
  rainbow(n, end=4/6, alpha=alpha)[n:1]
}
plot(psom, main = "cluster",palette.name = coolBlueHotRed)

soglia=0.0001
codes<-psom$codes
codes_norm<-matrix(nrow=64,ncol=8)
for(i in 1:8){
  codes_norm[,i]<-codes[,i]/max(codes[,i])
}
```

Figure 6.6. Code snippet to build the SOM network

output of the function is the code matrix, containing the weights assigned by the learning algorithm to each document. Figure 6.7 shows the lattice plotting that, similarly to the dendrogram for hierarchical analysis, is a way to visualize the result on a two-dimensional plan, allowing a monitoring of the process.

In the upper right of the lattice (see Figure 6.6) some SOM units containing similar documents from the two data sources can be noted. In particular, a cell containing the clustered documents for clients and customers can be observed. The same documents are clustered with the offices document. In the top left corner, there are clusters that contain the documents clients, customers, products and productLines as well as customers, payments, products and productLines.

Furthermore, it is noteworthy that in each cluster the content "segment" are proportional to the weight assigned to the document. Note also the distance between the SOM units arranged in the lower section of the pattern compared to those in the upper section. This suggests that between these SOM units, i.e., the documents they contain, there is strong dissimilarity.

However, as explained in the previous stage, the graphiocal representation of the lattice is a mere suggestion. In fact, a more detailed analysis of the output of the som function is needed. The weight matrix is normalized and items not exceeding the threshold set by the business manager, are discarded since they occurrence is limited in the SOM unit. The new the weight matrix is split by columns, then

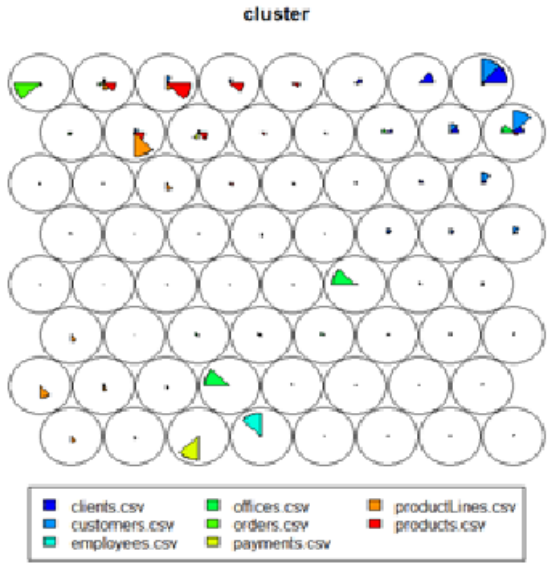


Figure 6.7. SOM lattice (64 unit SOM)

we calculate the covariances matrix for each pair of vector columns (each vector contains of 64 elements) obtaining a covariances matrix of cardinality 8 x 8. Figure 6.8 shows the code snippet to get the covariance matrix, covmat.

The covmat matrix is the input for the cluster analysis phase, which is quite similar to that carried out at he end of the syntactic analysis phase. Remember that the cluster analysis goal is the execution of the k-means algorithm to find well outlined clusters (minimum inter-cluster variance). Figure 6.9 shows the dendrogram of hierarchical clustering, showing the hierarchy of documents based on their semantic similarities.

Table 6.2 summarizes the execution of the algorithm for the sample use case. Documents are identified by the number assigned to it in Table 6.1. It is worth

```
...
codes_norm[which(codes_norm< soglia)]<-0
#split della matrice codes_norm
clients<- codes_norm[,1]
customers<- codes_norm[,2]
employees<- codes_norm[,3]
offices<- codes_norm[,4]
orders<- codes_norm[,5]
payments<- codes_norm[,6]
productLines<- codes_norm[,7]
products<- codes_norm[,8]

lista<-list(clients,customers,employees,offices,
            orders,payments,productLines,products)

#MATRICE DI COVARIANZA
covmat<-matrix(ncol=8,nrow=8)
colnames(covmat)<-ds$Names
rownames(covmat)<-ds$Names
   for(i in 1:8){
     for(j in 1:8){
        #sulle diagonali ho le varianze di ogni vettore
       covmat[i,j]<-cov(lista[[i]],lista[[j]])
     }
   }
...
```

Figure 6.8. Code snippet to derive the covariances matrix

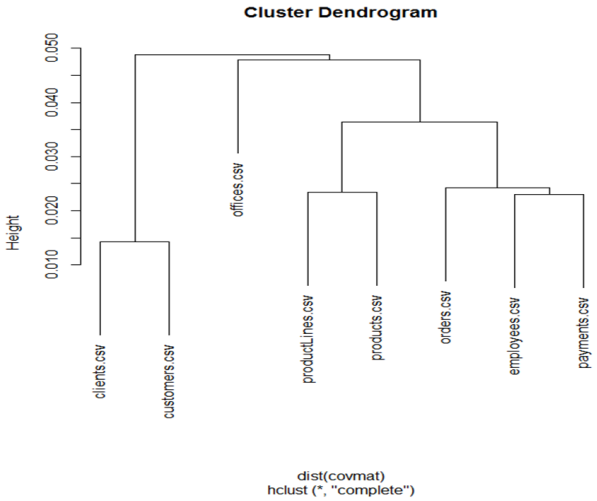


Figure 6.9. Semantic analysis dendrogram

noting that two executions of the algorithm are required, and the documents clients, customers, employees, orders, payments, productLines, and products are divided

into 3 clusters. Fitting value is good (76%). Even in this case, clusters C1, and C2 obtained in the second iteration suggest the existence of matching between the two source schemata.

6.3.5 Comparison of analysis results

At this stage of the process, a comparison between the results obtained from the syntactic and semantics analyses, can guide us towards the decision of the best approach to capture the possible matching between the sources of data. Since the input of the syntactic analysis is completely different from that of semantic analysis, an index that can provide the clustering goodness is needed. We use the $between\_SS/total\_SS$ value returned by the *kmeans* algorithm. The reason for this choice is also justified by the nature of the *kmeans* algorithm that requires a low external interaction (to set the k parameter) for cluster calculation. The two indices, however, are not comparable, since we cannot establish an ordering relationship. As so, is a linear combination of two values will be used. In particular, the comparison function is:

$F : \alpha * Max_{sint}(between\_SS/total\_SS) + \beta * Max_{seman}(between\_SS/total\_SS)$

where $\alpha, \beta \geq 0$, $\alpha + \beta = 1$, Max_{sint} is the best *kmeans* value for syntactic analysis, and Max_{seman} is the best *kmeans* value for semantic analysis.

Parametri α e β are chosen by the business manager who, even assuming no knowledge about the algorithm, can tune them by displaying the dendrograms plotted after the analyses as well as the lattice of the SOM. Generally, semantic analysis is considered more exhaustive and efficient, so to give it an higher weight, $\beta > \alpha$.

In our example, the indices $between\_SS/total\_SS$ obtained from the analyses are respectively:

- 83%

- 76%

An appropriate choice of parameters could be $\alpha = 0.4$ e $\beta = 0.6$ thus slightly preferring semantic analysis. Looking at Table 6.2 we found that correlations exist between the two sources highlighted by the following clusters:

- C1=clients, customers
- C2=employees, orders, payments

Note that tables in cluster C1 describe exactly the same domain of interest, that is, the customers of the Car Shopping company. In fact, they effectively represent the integration of the two sources. As a consequence, the two tables will be the input of the future stages of the process. Note that the cluster C1 is also present in the output of the syntactic analysis.

6.3.6 Data cleansing

This phase of the process aims to ensure the consistency of the data contained in the tables and, as a consequence, from the documents to be integrated. In computer science, the data cleansing, or cleaning, or scrubbing, is the process for determining and correcting (or possibly remove) corrupted or inaccurate records from a set of records, tables or databases. The goal is twofold: trying to unify the schemes underlying the data (with obvious advantages, for example, in terms of flexibility and usability); trying to detect "atomic" (i.e., isolated or non-systematic values) errors like, for example, types (i.e., misspelling).

In practice, various correction techniques have been proposed, that can summarized in two approaches:

schema-level approach, that tries to establish a match between different file or database structures, taking advantage of specific similarities. This approach

involves manipulating the source schemata in order to define the reconciled scheme. For example, to merge tables t1, t2 having the following schemata:

- $t1(id, firstname, lastname, city)$
- $t2(pid, Name, Surname, address, city)$

we need to establish the correspondences:

- $id \rightarrow pid$
- $(name, surname) \rightarrow (Name, Surname)$
- $city \rightarrow (city)$

to obtain the reconciled schema containing data from both sources:

- $t12(pid, Name, Surname, address, city)$

instance-level approach, that tries to calculate a metric to measure the similarity, for example, among the records in the tables, or considering individual attribute values.

Instance-level data cleansing can generally be applied during population of the the data warehouse from the data sources (e.g., using standardized values from a specific vocabulary).

In our example, a schema-level data cleansing was performed. Indeed, by taking advantage of the output of the previous phase, i.e., the clusters computed by the k-means algorithm, we tried to find the correspondences between the columns in the tables contained in the clusters. At this stage, the tables are considered as a whole, thus, columns with numeric attributes will not be subjected to the data cleansing phase, but they will be maintained in order to support the merging phase of the next step of the process. Also, if necessary, the table schema syntax is modified to resolve ambiguity.

Therefore, the data cleansing phase aims at building a correspondence matrix between table columns to be merged. As usual, in our example I consider the integration of two different tables from their respective data sources, however it is possible to extend the idea to a greater number of tables and data sources.

Remember that a cluster contains at least two tables. The correspondence matrix will have cardinality m x n where m and n are the number of columns in the two tables (referred to as S and T in the following) of the input cluster. At the intersection of the m^{th} row and the n^{th} column in the matrix, there will be a value showing the result of the intersection operation over all possible pairs of columns coming from the two tables. In detail, for each pair of columns (x, y), where $x \in S$ and $y \in T$, the intersection function is applied. This function returns the common elements of the two sets of values, extracted by a comparison between the single columns of S and T. For intersections between strings, the comparison is based on the alphabetic ordering of the strings. We excluded from this operation columns containing only numeric values, because they may be measures or indexes that we do not want to modify. To strengthen this motivation, imagine two tables to be integrated (e.g., clients and customers), and consider the existence of column pairs (age for clients, weight for customer). The fusion of these two columns is misleading for the different semantic nature of the columns themselves. In fact, age and weight are two metrics that measure time and mass, respectively.

From the correspondence matrix, for each row (or column) the highest value is considered. The corresponding column (or row) of the table represents the column to be integrated with that row (or column). Figure 6.3 shows an example of correspondence matrix for tables S and T, where S = [x, y, z] and T = [a, b, c, d]. The matrix is created by considering the attributes of S (the table with the lowest number of attributes) on the rows.

It can be noted from the correspondences, that the pairs of columns to be integrated are (x, c); (y, b); (z, d) considering the maximum value for each line.

Table 6.3. Correspondence matrix for tables S and T

		Table T			
		a	b	c	d
Table S	x	0	0	**7**	0
	y	0	**12**	0	0
	z	0	2	1	**16**

The attributes of the identified pairs are stored in two different lists (related to tables S and T) that will be the input for the merging phase. Such attributes will be considered key attributes and will be used to perform the merging. It is important to note that an inaccurate match can be obtained, causing loss of information due to the nature of the intersection function. In fact, in case of data with repetitions this function tends to delete duplicates. As a consequence, it may happen that a whole record is not included because in the sources there is another record having the same values in key fields (used by the merge).

In our example, to compute the correspondence matrix, we used the intersect (x, y) function of the R base package that discards duplicate values in the arguments, and apply them to the *as.vector* function. Figure 6.10 shows the matrix obtained for cluster C1 of the previous phase containing client and customers tables.

row.names	customerNumber	customerName	contactLastName	contactFirstName	phone	...
clientId	0	0	0	0	0	
clientName	0	122	0	0	0	
clientPhone	0	0	0	0	121	...
clientAddress	0	0	0	0	0	
officesEmployeeNumber	0	0	0	0	0	...

Figure 6.10. Sample correspondence matrix

Merge will taking into account the following keys:

- $clientName, clientPhone$ for clients;
- $customerName, phone$ for customers.

Other columns are omitted because they have all *null* values. The final matrix cardinality is 5 x 13.

6.3.7 Data source Integration

This phase is intended to integrate the source tables and produce a file containing the reconciled data. To this aim, the join operation of relational databases can be used. In particular, the *innerjoin* operator creates a new table by combining the values of two input tables (e.g., A and B) based on a specified comparison rule. When the join rule is met, the values of all the columns from tables A and B are combined in a single row of the resulting table. On the other hand, the *outerjoin* operator does not require exact correspondence between the rows of two tables. The table resulting from an outer join holds even records that have no match between the tables. The *outerjoin* can be specified into *left outer join*, *right outer join*, and *full outer join*, depending on which one the table has to hold the values in case of non-matching. In this case left and right refers to the sides of the JOIN keyword.

In our process the *inner join* and the *full* (*left* and *right*) *outer join* are used. The keys used to determine the join are contained in the lists produced by the previous phase of data cleansing. These lists contain pairs of keys, where the i^{th} key from table S is paired with the j^{th} key from table T.

As an example, consider the process of data cleansing of our example data sources. The output of this step will be a table containing the common columns of the two source tables (showed in blue in Figure 6.11). Attributes in A not matching in B are shown in green, while attributes in B not matching in A are shown in red.

Once you've added the tables you've found, go to the last step of the first one step of the process, that is, the construction of the reconciled scheme.

If we apply the cleansing phase to S (the clients table) and T (the customers table), we obtain two lists, $keys\_S$ and $keys\_T$. The *Merge* implemented in R

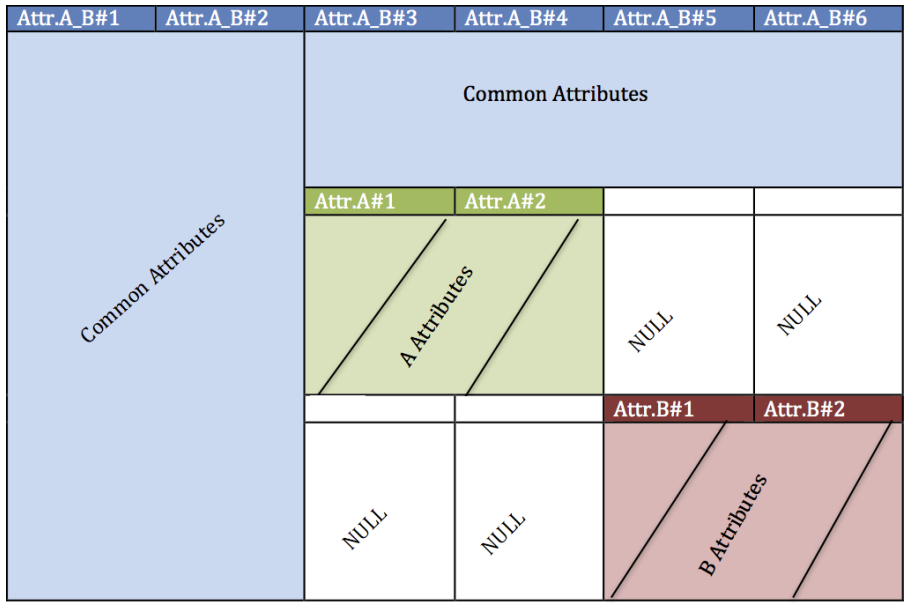


Figure 6.11. Integrated table. Common attributes are shown in blue, attributes in A not matching in B are shown in green, while attributes in B not matching in A are shown in red

is then used to perform the merging of the common columns between S and T considering the key lists for S and T and keeping the remaining columns in both S and T. By setting the *all* parameter of the merge function we can obtain different types of joins. In particular:

- $all = FALSE \rightarrow$ natural join;
- $all.x = TRUE \rightarrow$ left (outer) join;
- $all.y = TRUE \rightarrow$ right (outer) join;
- $all = TRUE \rightarrow$ full outer join.

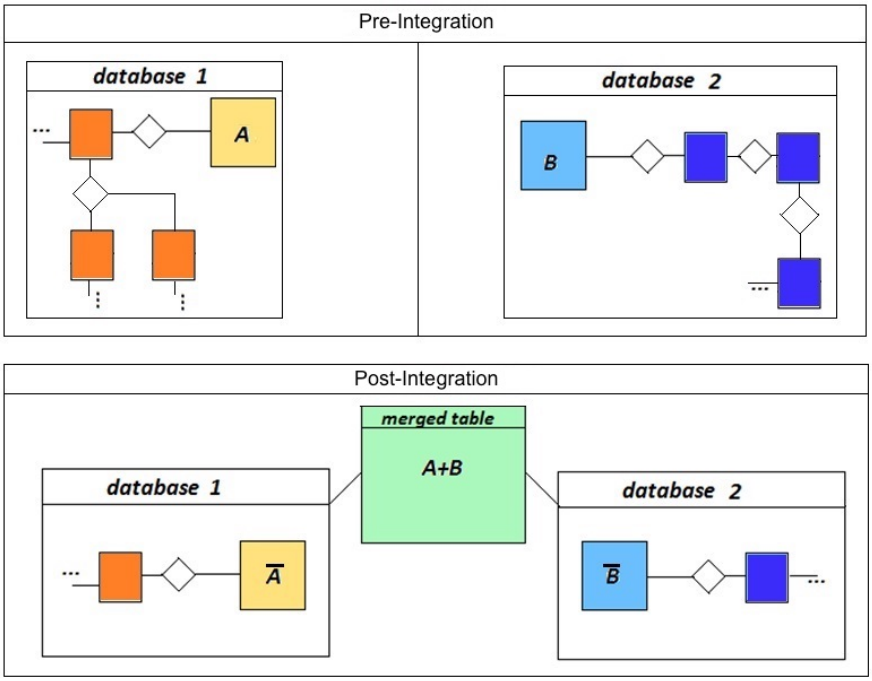


Figure 6.12. Schema integration

6.3.8 Reconciliation scheme

The final step of the data integration is to build the reconciled schema of the sources. The final data warehouse will have a three-tiered architecture (see Figure 6.13).

In the proposed process, reconciled data reside in a new relational database containing the integrated tables produced during the previous step, the remaining non-integrated sources, together with new concepts deriving from semantic properties correlations. Figure shows the reconciled schema for the model illustrated in Figure .

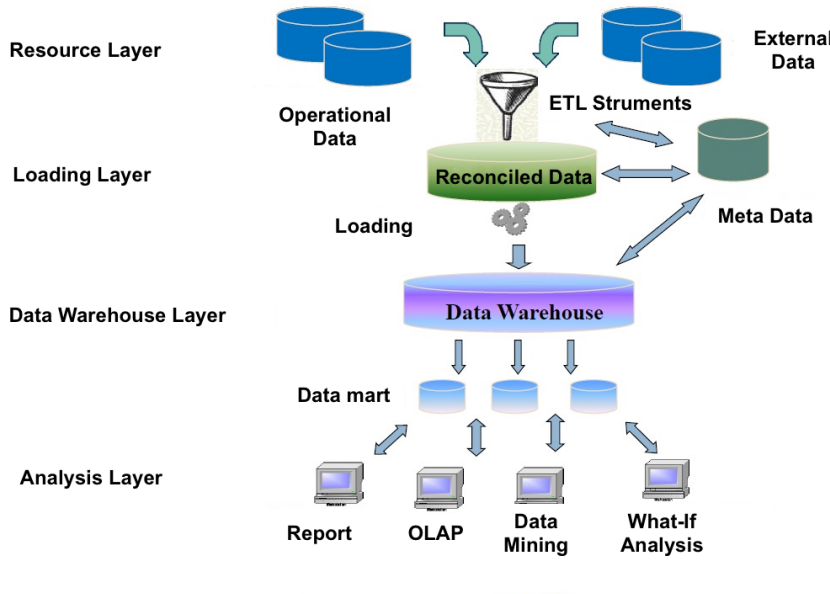


Figure 6.13. Three-tier data warehouse architecture

6.3.9 A special case

In case it is necessary to integrate a number of sources greater than two and therefore there is a semantic similarity between a number of tables greater than two. If this scenario occurs, the data integration of the sources is carried out following a balanced binary integration strategy. This strategy starts with a first pair of sources that will be integrated, after which the dataframe (the integrated table) obtained will be integrated to the result of the merging of another possible pair of semantically similar tables (see Figure 6.15). The results will be merged and the process will be repeated until we have a single integrated table. If the number of semantically similar tables to be integrated is even, the process is straightforward. If the number of tables to be integrated is odd number, at least one table will be replicated twice and analyzed with the corresponding semantically similar table.

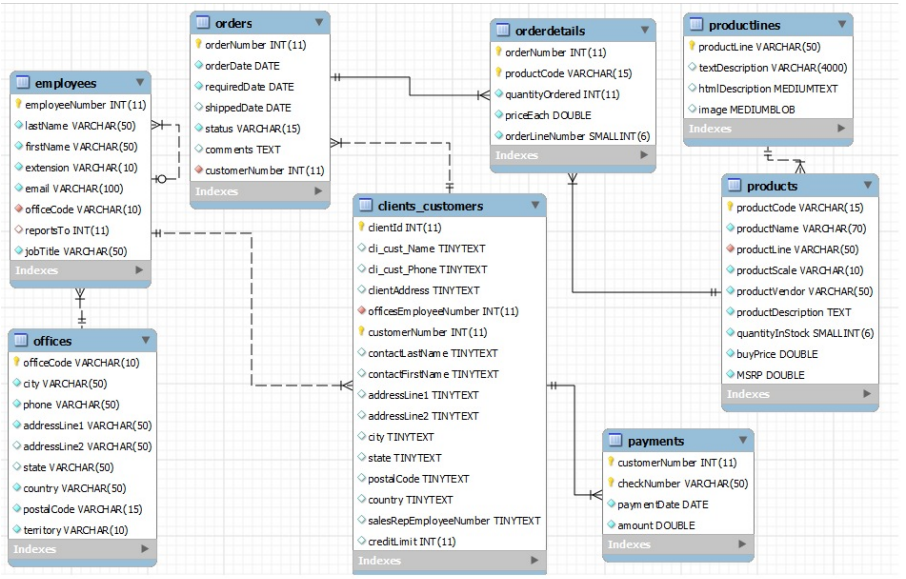


Figure 6.14. Reconciled schema

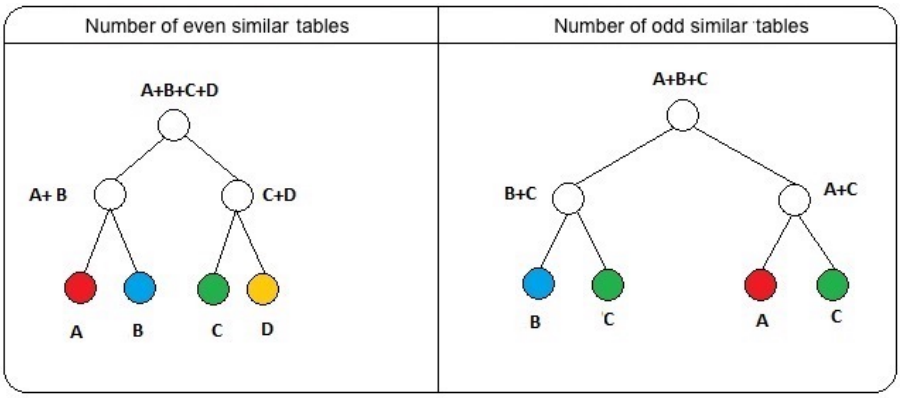


Figure 6.15. Intefgration strategy in case of even (left side) or odd (right side) tables to integrate

In the first case, semantic similarities between the tables (A, B) and (C, D) are detected, while in the second case semantic similarities between tables (B, C) and

(A, C) are detected. It is evident that in case of integration of an odd number of tables (e.g., A, B and C), there is data redundancy and therefore an inevitable slowdown in the merging process.

6.4 OLAP Schema Generation

The goal of the second step is the construction of the Data Warehouse and the visualization of the data it contains to support decision-making processes. This step was designed to support managers without technical skills. Then, from the perspective of data integration, this step aims to expose the results obtained through an intelligent construction of the cube of facts in order to perform strategic OLAP analysis, using Information Retrieval techniques for processing the business rules dictated by the manager.

Starting from the integrated scheme, realized through the first step of the proposed process, and from the business rules (expressed in natural language), the proposed semi-automated procedure enabled the generation of OLAP schemas for the data mart that will populate the data warehouse. This is accomplished by performing the conceptual design, the logical design and the final generation of the data warehouse.

6.4.1 Conceptual design of the Data Warehouse

The Data Warehouse is constructed in a completely automatic way based on the reconciled relational database produced in the previous step. In section I present the activities to be carried out in order to complete the first step in the generation of the OLAP scheme of a Data Warehouse, i.e., the conceptual design phase of the Data Mart used to populate the Data Warehouse. In particular, the design and construction steps of the DFM will be described starting from the reconciled

scheme. The proposed technique for the conceptual design of a Data Mart, according to the DFM, consists of the following steps:

- selection of the facts;
- attribute tree construction;
- definition of dimensions / measurements;
- creation of the fact scheme.

6.4.1.1 Selection of the facts

To obtained a selection of facts that conforms to the company's strategic needs, a procedure based on the set of concepts returned by the analysis of the data in relation to the business rules is executed. The procedure consists of the following steps:

- defining the set of candidate entities to become a fact;
- application of a metric for the choice of facts based on the set obtained in the previous step.

The success for the construction of an efficient and useful Data Warehouse is the Information Retrieval (IR) phase. This is conducted to analyze the business rules imposed by the manager/decision maker, in order to focus the analysis on strategic decisions for the company's business. The identification of the relevant concepts within the data according to the business rules is not easy to realize. Indeed, the latter are expressed in natural language.

In order to simplify the analysis of data based on the business rule, this automated phase is made through an Information Retrieval technique, in particular the LSI (Latent Semantic Indexing) technique is used.

For the execution of LSI, we need, for each table of the reconciled schema, a document containing all its tuples (table document) and for each business rule of a document that contains its text (rule document). The content of each document must undergo a normalization step in which (i) the non-textual terms are eliminated (e.g., operators, special symbols, numbers, etc.), (ii) the words composed of two or more words are subdivided (for example, "mail_address" is transformed into "mail" and "address") and (iii) terms with a length of less than two characters are not removed. Furthermore, a stemming algorithm is also used to reduce derivative terms to their common root. In LSI, documents represented as points (vectors) in an m-dimensional space, where m is the number of documents, are represented in a k-dimensional space, with $k < m$. In this way, every document, table and business rule, has its representation in the k-dimensional space. The set of vectors of the k-dimensional space, represents in a certain sense the set of concepts or different meanings that the various documents contain. Using the similarity cosine as a measure of similarity for calculating the proximity of the documents in the k-dimensional space, we obtain a matrix of similitude coefficients for each possible combination of documents.

For each document that represents a business rule (rule document), we identify the documents that represent a table (table document) having the highest value for the similarity cosine. In case this value exceeds a certain threshold set by the manager, we can take in into consideration [1]. The name of the table represented by the document identifies a concept of interest for the strategic needs and is therefore essential for the construction of the Data Warehouse.

In this way, given the business rules, it is possible to retrieve the set of concepts of interest to build the facts on. These objects will be used by the decision-maker

[1] Note that a certain percentage of error exists, depending on the adopted similarity threshold (the higher the threshold, the lower the error). Unfortunately, high thresholds reduce the ability of the technique to identify similarities.

of the organization. In other words, for each business rule, the set of documents pertaining to the rule is identified within a certain threshold. Subsequently, a further analysis step retrieves the documents that occur in the sets obtained in the previous step, and having a similarity cosine value higher than a certain threshold imposed by the manager. The candidate table to become the fact will be the table with the highest frequency of similarity cosine values close to 1. In formal terms, the F fact table is such that:

$F = max\{count(cos(br_i, doc_j) \cong 1)\}$

where i is the number of rules documents and j is the number of table documents.

For our sample case study, available business rules are:

- BR1) If it is necessary to acquire new professional figures, what professional roles would be required?
- BR2) What are the employees involved in orders for each product line?
- BR3) Is it necessary to upgrade the service to increase the number of orders managed in certain business areas?

By applying the described procedure, for step 1, the candidate facts entities are E = Products, Orders, ProductLines. In fact, they are the entities reporting the maximum cosine similarity value for the three business rules mentioned above. Figure 6.16 shows the similarity matrix computed by LSI.

Regarding the metric to be chosen in the second step of the procedure, a similarity threshold greater than or equal to 95% is considered. Usually, 5% residual errors are admissible in statistics. This assertion is due to a property of the standardized normal distribution, also called an accidental error curve (i.e., errors occur randomly and are independent of each other). Informally, accidental errors are unavoidable in the measures. However, their effect can be reduced by repeating the

row.names	clients_customers.csv	employees.csv	offices.csv	orderdetails.csv	orders.csv	payments.csv	productLines.csv	products.csv
clients_customers.csv	1.000000000	-9.989925e-01	0.99729937	4.487804e-02	-0.01109219	0.6458455	0.2285997	0.1873760
employees.csv	-0.998992473	1.000000e+00	-0.99959057	-2.563477e-16	-0.03379427	-0.6794577	-0.2720591	-0.2312704
offices.csv	0.997299373	-9.995906e-01	1.00000000	-2.861275e-02	0.06237684	0.7001731	0.2994812	0.2590127
orderdetails.csv	0.044878041	-2.563477e-16	-0.02861275	1.000000e+00	-0.99942881	-0.7337147	-0.9622805	-0.9728895
orders.csv	-0.011092188	-3.379427e-02	0.06237684	-9.994288e-01	1.00000000	0.7562574	0.9709249	0.9801494
payments.csv	0.645845459	-6.794577e-01	0.70017312	-7.337147e-01	0.75625736	1.0000000	0.8908920	0.8709618
productLines.csv	0.228599720	-2.720591e-01	0.29948120	-9.622805e-01	0.97092494	0.8908920	1.0000000	0.9991119
products.csv	0.187375987	-2.312704e-01	0.25901273	-9.728895e-01	0.98014943	0.8709618	0.9991119	1.0000000
query1.txt	0.362804192	-4.042590e-01	0.43026394	-9.146446e-01	0.92778378	0.9457650	0.9901270	0.9833412
query2.txt	-0.001594718	-4.328487e-02	0.07185309	-9.990628e-01	0.99995490	0.7624373	0.9731547	0.9819882
query3.txt	-0.021108881	-2.378043e-02	0.05237535	-9.997172e-01	0.99994982	0.7496650	0.9684781	0.9781141

Figure 6.16. Similarity matrix of the sample case study

measure of a quantity several times and considering accidental error as a casual random variable. This property asserts that 95% of the values of an observation is between the mean and ±1.96 standard deviations, as shown in Figure 6.17.

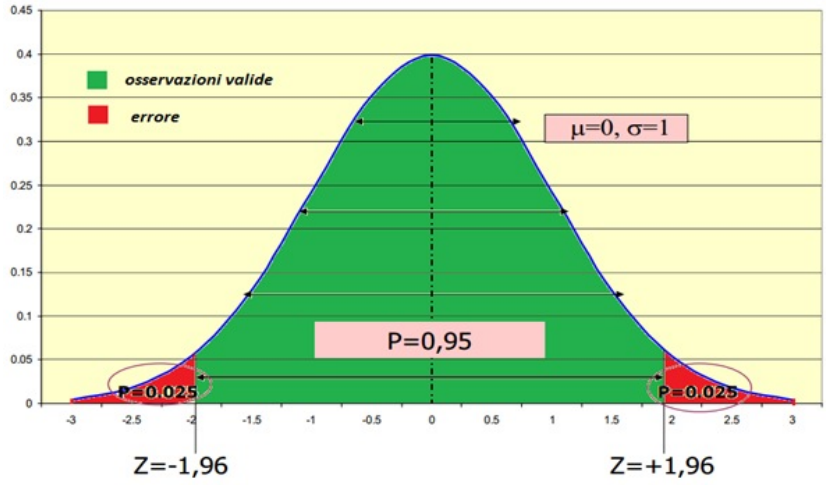


Figure 6.17. Error curve

In Figure 6.16, the values of the similarity cosine are shown in red for the table documents (by columns) which have a minimum value of 0.95 calculated as a measure of similarity with the three business rules defined before. Remember that the threshold is chosen by the user of the system. It is worth noting that the error made in identifying the candidate tables to represent a fact, given the

business rules, is lower than 0.05 (i.e., 5%). Looking at the figure and considering the formula used to determine the fact table, we note that the entity *Orders* is the entity that has almost 100% similarity with BR2 and BR3, so it is the entity chosen as a fact.

6.4.1.2 Attribute tree construction

Given an entity F designated as fact, we define the *attribute tree* a structure that meets the following requirements:

- Each vertex corresponds to a simple or compound attribute of the source schema;
- The root corresponds to the F identifier;
- For each vertex v, the corresponding attribute functionally determines all the attributes that correspond to the descendants of v.

Figure 6.18 shows the algorithm used for the attribute tree construction. The input is represented by the reconciled scheme of Figure 6.4.3.3. F is the selected fact (i.e., Orders). Figure 6.18 reports the attribute tree for the Orders fact. A pruning was applied by deleting descriptive and optional attributes.

6.4.1.3 Definition of dimensions and measures

The dimensions determine how events can be meaningfully aggregated for decision making. They must be chosen in the tree of attributes between the vertices children of the root. They have to correspond to discrete attributes or to discrete or continuous intervals.

Time is a key factor in the design of a Data Warehouse, because it is important to conduct analysis involving time intervals. To this end, the time must be a

```
root=nuovoVertice(ident(F)); /* ident(F) è l'identificatore di F,
                             la radice dell'albero è etichettata
                             con l'identificatore dell'entità
                             scelta come fatto */

traduci(F,root);

procedura traduci(E,v);
/*E è l'entità corrente dello schema sorgente, v il vertice corrente
dell'albero */
{
  per ogni attributo a di E tale che a!=ident(E)
   aggiungiFiglio(v, nuovoVertice(a))//aggiunge al vertice v un figlio a
  per ogni entità G connessa ad E da una associazione R tale che
  max(E,R)=1
   {
     per ogni attributo b di R
          aggiungiFiglio(v, nuovoVertice(b));
          //aggiunge al vertice v un figlio b
     prossimo = nuovoVertice(ident(G));
     //crea un nuovo vertice con il nome dell'identificatore di G ...
     aggiungiFiglio(v, prossimo);// … lo aggiunge a v come figlio …
      traduci(G, prossimo);
```

Figure 6.18. Algorithm used to build the attribute tree

dimension of the Data Warehouse, and if a temporal dimension is not defined for the scheme, it can be added by navigating the attribute tree [44].

Concretely, in the tree, we identify the attributes of a given type that are candidates for becoming dimensions; if these attributes are not direct children of the root, they will surely be reachable by means of joins of the source schema tables, and so they can be easily transformed into direct children of the root, and therefore dimensions of the future multidimensional schema.

A measure is a numerical property of a fact and describes a quantitative aspect of interest for analysis. Therefore, the measures must correspond to numerical attributes, to which aggregation functions can be applied (sum, average, maximum,

minimum) [44]. Regarding the identification of the measures, we can reason similarly to dimensions: if a measure is not a numeric attribute direct child of the root that identifies the fact, it is possible to reach it by joining the source schema and then adding that measure to the fact table.

In Figure 6.18, the date type *orderDate* and *requiredDate* are already direct children of the root *orderNumber*, and so they are already two dimensions in a direct way. In our sample case study, we add as measures all the attributes of the candidate tables to become a fact of the multidimensional scheme that were identified by the LSI technique. In particular, *orders* is the fact, the other tables identified by the LSI technique are *products* and *productLine*. Thus, their attributes (excluding primary keys) will all become measures of the actual scheme. If a measure, and therefore the attribute it represents, does not have a numerical domain, it makes no sense to perform aggregation functions on it. For the purpose of the automatic construction of the Data Warehouse, such a non-numerical measurement is aggregated by the "max" operator which simply returns the value of the attribute. In our example, the measures will then be the attributes of the *orders*, *products* and *productLine* tables. Table 6.4 shows the measurements of the identified multidimensional scheme and the tables of the source scheme from which the attributes originate. The last three rows show the measurements coming from the *orderdetails* table of the source schema as they are considered useful for the fact orders (see Figure 6.4.3.3).

6.4.1.4 Creation of the fact scheme

The attribute tree can be translated into a schema that includes the dimensions and the measures defined [44]. The name of the fact corresponds to the name of the entity to which the attribute chosen belongs as representative of the fact. The hierarchies correspond to the subtree trees of the attribute tree with root in the various dimensions. Figure 6.19 shows the diagram for the orders fact. Note the

Table 6.4. Attributes used to identify fact measures

Attribute	Source table
status	orders
productName	products
productScale	products
productVendor	products
quantityInStock	products
buyPrice	products
MSRP	products
productline	productLine
quantityOrdered	orderdetails
PriceEach	orderdetails
OrderLineNumbe	orderdetails

presence of multiple links for relations (*: N) existing on the attribute tree, as well as the presence of the recursive hierarchy *employeeNumber* → *reportsTo* → *EmployeeNumber*. Also *productLine*, the only attribute of the *productlines* table can be cropped because it is an external key already in the *products* table.

By creating the actual schema, the Data Warehouse conceptual design phase ends. Note that all the steps in this phase have been sequentially and automatically carried out.

6.4.2 Logical design of the Data Warehouse

This section describes the steps to be taken in order to complete a further step toward the generation of the OLAP schema of the Data Warehouse, i.e., the logical design of the Data Mart that will be used to populate the Data Warehouse. Specifically, the steps to transform the conceptual scheme, obtained in the previous phase, in a corresponding logical scheme will be presented. There are two distinct logical models to represent the multidimensional structure some data:

- the relational one, which produces ROLAP systems;

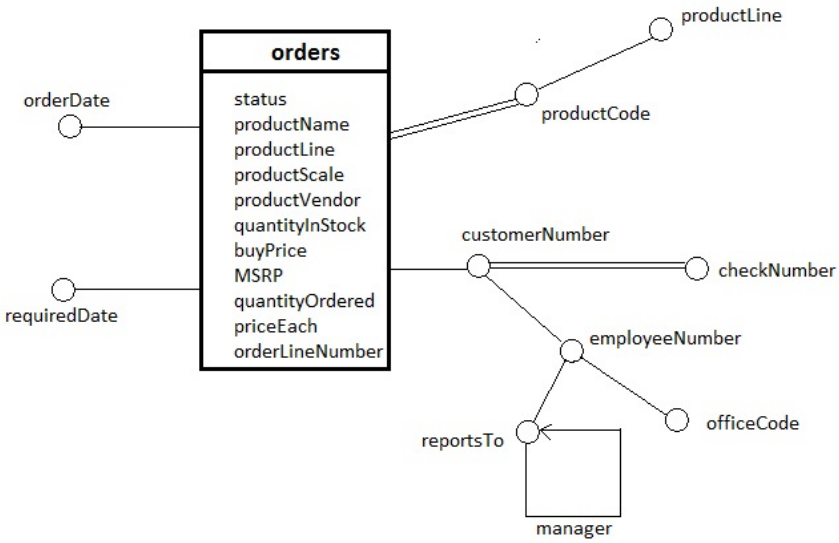


Figure 6.19. Scheme for the *orders* fact

- the multidimensional one, which produces MOLAP systems.

A ROLAP system will be used, because a set of problems related to MOLAP systems, such as data sparseness or lack of standards for querying data, make it difficult to use. Furthermore, there are additional reasons for the adoption of a two-dimensional model for modeling multidimensional concepts:

- The relational model is the "de facto" standard of databases, and is known by industry practitioners;
- The evolution of relational DBMS, for thirty years on the market, makes them refined and optimized tools;
- The lack of data gap guarantees greater scalability, crucial for ever-growing databases such as Data Warehouses.

For multidimensional modeling on ROLAP systems, the star schema is used. The star schema consists of:

- a set of relationships $DT1, ..., DTn$, called dimension tables, each associated with a size and characterized by a primary key and a key set of attributes that describe the dimension at various aggregation levels;
- An FT relation, called fact table, whose primary key is the set of the primary keys of the dimension table. The FT relation has an attribute for each measure.

From a star scheme, a snowflake scheme can be obtained by decomposing one or more dimension tables DTi in multiple tables $DTi_1, ..., DTi_n$. Each resulting dimension table is characterized by:

- A primary, key di, j (usually surrogate)
- A subset of DTi attributes that depend on di, j
- Zero or more external keys related to other $DTtk$ required to secure the rebuildability of DTi's information content

The primary dimension tables are those whose keys are imported into the fact table, while the others are secondary. Remember that a hierarchy that encodes any-to-many type associations is modeled with multiple arcs. At the logical level there are several solutions for the management of multiple arcs. *Bridge solution*: use of a new table whose key is composed of the combination of the attributes connected by the multiple arc (a snowflake pattern is obtained). *Push-down solution*: the any-to-many association is modeled directly within the fact table. A new dimension is then added corresponding to the terminal attribute a of the multiple arc, and any children of a will be stored in the new dimension table (a star schema is maintained).

The push-down solution introduces a strong redundancy in the fact table whose lines must be repeated as many times as there are multiple-arc matches. The strong redundancy causes very expensive update operations, which does not occur if the bridge table solution is used.

For these reasons, the bridge-based approach is adopted as a solution, obtaining a logical snowflake scheme.

Figure 6.20 shows the logical snowflake scheme for the *orders* fact. With the creation of the snowflake logical scheme, corresponding to the DFM for the identified fact, the logical design phase of the Data Warehouse ends. The next phase will be the concrete realization of the Data Warehouse.

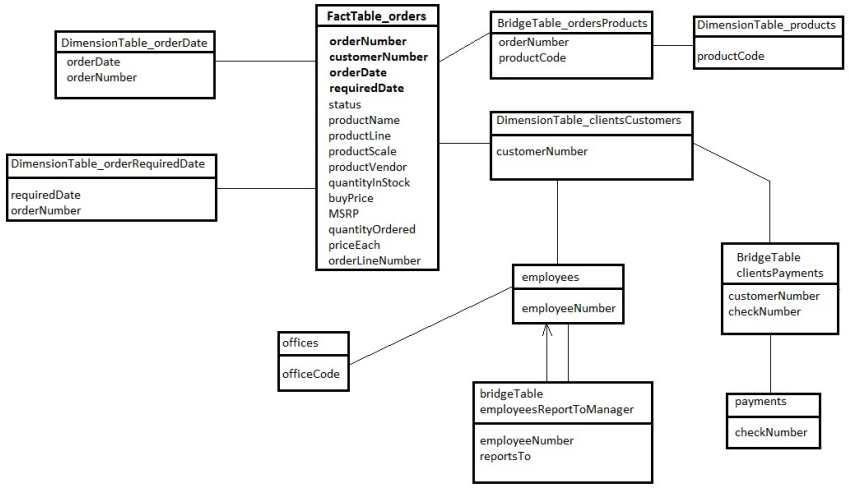


Figure 6.20. Logical snowflake scheme for the *orders* fact

6.4.3 Implementation of the Data Warehouse

This section describes the steps to be taken in order to concretely generate the OLAP schema of the Data Warehouse. In particular, two steps are described. In the first step we define the SQL code for the creation of the Data Warehouse. In

the second step an XML file is created for the Data Mart. This file describes the multidimensional cube structure and is necessary for the Pentaho Mondrian engine because it requires that the description of the multidimensional schema of the data foundation be provided as an XML file.

6.4.3.1 SQL database

The following is a fragment of the SQL script for creating the Data Warehouse (Figure 6.21). Figure 6.22 shows the snowflake ER diagram.

6.4.3.2 Mondrian schema

The multidimensional schema of the Data Warehouse should be mapped to an XML file [45], because the Mondrian engine[2] requires that the description of the multidimensional database schema be provided in the form of an XML file.

This file can be edited manually or using the tool used in the Pentaho suite (i.e., Schema Workbench). Figure 6.23 shows a fragment of the XML file for the orders_cube cube.

The <Schema> tag is used to define a multidimensional database that contains a logical model, consisting of cubes, hierarchies and their members, and a mapping of this model based on a physical model that is the source of the data, i.e., the ROLAP system implemented to the previous phase, which is presented through the logical model.

In our example, the logical schema is called *schemaClientCustomers* and consists of a cube orders_cube, as indicated in the <Cube> tag, which specifies a collection of dimensions and measures.

The <Table> tag indicates that the orders_cube cube is mapped to the fact

[2]the component for managing the multidimensional cube and for retrieving data.

```
CREATE DATABASE  IF NOT EXISTS `dw_clientcustomer` /*!40100 DEFAULT CHARACTER SET latin1 */;
USE `dw_clientcustomer`;

CREATE TABLE `join-clients_payments` (
  `checkNumber` varchar(50) NOT NULL,
  `customerNumber` int(11) NOT NULL,
  KEY `u` (`customerNumber`),
  KEY `p` (`checkNumber`),
  CONSTRAINT `u` FOREIGN KEY (`customerNumber`) REFERENCES `clients_customers` (`customerNumber`)
  ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `p` FOREIGN KEY (`checkNumber`) REFERENCES `payments` (`checkNumber`)
  ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `join_orders_products` (
  `orderNumber` int(11) NOT NULL,
  `productCode` varchar(15) NOT NULL,
  KEY `q` (`orderNumber`),
  KEY `q2` (`productCode`),
  CONSTRAINT `q` FOREIGN KEY (`orderNumber`) REFERENCES `orders` (`orderNumber`)
  ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `q2` FOREIGN KEY (`productCode`) REFERENCES `products` (`productCode`)
  ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

DROP TABLE IF EXISTS `orders`;
CREATE TABLE `orders` (
  `status` varchar(15) NOT NULL,
  `productName` varchar(70) NOT NULL,
  `productLine` varchar(50) NOT NULL,
  `productScale` varchar(10) NOT NULL,
  `productVendor` varchar(50) NOT NULL,
  `quantityInStock` smallint(6) NOT NULL,
  `buyPrice` double NOT NULL,
  `MSRP` double NOT NULL,
  `orderNumber` int(11) NOT NULL,
  `customerNumber` int(11) NOT NULL,
  `orderDate` date NOT NULL,
  `requiredDate` date NOT NULL,
  `order_id` int(11) NOT NULL AUTO_INCREMENT,
  `quantityOrdered` int(11) NOT NULL,
  `priceEach` int(11) NOT NULL,
  `orderLineNumber` smallint(6) NOT NULL,
  PRIMARY KEY (`order_id`,`orderNumber`),
  KEY `d1` (`customerNumber`),
  KEY `d2` (`orderNumber`),
  CONSTRAINT `d1` FOREIGN KEY (`customerNumber`)REFERENCES `clients_customers` (`customerNumber`)
  ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB AUTO_INCREMENT=2997 DEFAULT CHARSET=latin1;
```

Figure 6.21. SQL script fragment for the DW creation

table named *orders*. Then the cube dimensions are declared through the <Dimension> tag.

The foreign key attribute of the <Dimension> tag indicates that the name specified for this attribute is an external key that refers to the name of a column in the fact table.

The <Hierarchy> tag indicates a set of levels (members of the hierarchy) organized into a structure to facilitate analysis. The name attribute specifies the name

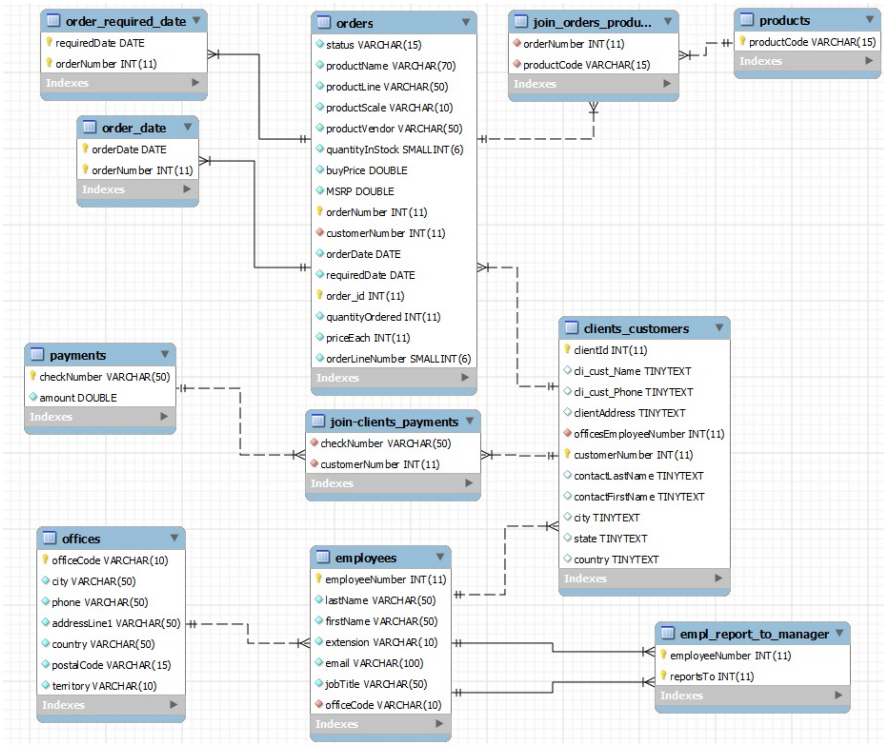


Figure 6.22. Snowflake schema

used in MDX queries to refer to that hierarchy, the primaryKey attribute specifies the primary key name of the dimension. In addition, if the hierarchy contains more than one table, ambiguities can be eliminated by using the primaryKeyTable attribute, that can be used to specify the table representing the primary dimension.

If specified as the child of the <Hierarchy> tag, the <Table> tag is used to indicate that the size is mapped to the table indicated by the name attribute.

In the case of a snowflake schema, the <Join> tag is used to connect the tables in which the original dimension table is decomposed. Attributes are specified for that tag leftKey and right Keys that specify the keys needed to join in two tables involved.

```
<Schema name="schemaClientCustomers">
  <Cube name="orders_cube" visible="true" cache="true" enabled="true">
    <Table name="orders">
    </Table>

  <Dimension type="StandardDimension" visible="true"
  foreignKey="orderDate" highCardinality="false" name="dataOrdinazione">
    <Hierarchy name="dateOrdinazione" visible="true"
    hasAll="true" allMemberName="Date_di_ordinazioni">
      <Table name="order_date">
      </Table>
      <Level name="DataOrdine" visible="true" table="order_date"
      column="orderDate" type="String" uniqueMembers="false" levelType="Regular"
      hideMemberIf="Never">
      </Level>
    </Hierarchy>
  </Dimension>
     ...
    <Dimension type="StandardDimension" visible="true"
    foreignKey="orderNumber" highCardinality="false" name="prodotti">
      <Hierarchy name="Prodotti" visible="true" hasAll="true"
      allMemberName="tutti_Prodotti" primaryKey="orderNumber"
      primaryKeyTable="join_orders_products">
        <Join leftKey="productCode" rightKey="productCode">
          <Table name="join_orders_products">
          </Table>
          <Table name="products">
          </Table>
        </Join>
        <Level name="prodotto" visible="true" table="products"
        column="productCode" type="String" uniqueMembers="false"
        levelType="Regular" hideMemberIf="Never">
        </Level>
      </Hierarchy>
    </Dimension>
      ...
    <Measure name="NumeroLineaOrdine" column="orderLineNumber" aggregator="distinct-count"
     visible="true">
    </Measure>
  </Cube>
</Schema>
```

Figure 6.23. XML file fragment for *orders_cube*

The <Join> tag takes two operators that can be:

- <Table> tag elements to indicate the tables involved in the join operation;
- <Join> tags to nest a new join operation into the current one, thus allowing hierarchies on more than two tables.

In the second case, in order to make the consecutive joins involving more than two tables, the XML file representing the cube has to be manually edited, because Schema Workbench does not allow this operation.

The hierarchy levels are specified using the <Level> tag, which specifies the *name* attribute that indicates the name of the level, the *table* attributes (if the dimension concerns multiple tables) and *columns* that specify the table and column from which to read the key of the layer. The *uniqueMembers* attribute is a boolean

that is used for SQL optimization. In Figure 6.23, each dimension, together with its hierarchy, join, and layers are enclosed by a red rectangle (joins it in an orange rectangle).

At this point we can declare the fact measurements through the <Measure> tag. For each measure, we can specify the *name* attribute to indicate the name of the measure, the *column* attribute to indicate the corresponding column in the fact table, and the *aggregator* attribute to indicate the aggregation operator to perform on the measurement. This is highlighted by a blue rectangle in figure 6.23.

After this last phase, the Data Warehouse was concretely created. So, the cube can be published on the Pentaho server and we can start OLAP analysis using JPivot.

6.4.3.3 Population of the Data Warehouse

The Data Warehouse population requires the definition of necessary procedures for uploading data within the Data Mart from operational sources.

The data stored in the sources are extracted and cleaned with ETL tools. The tool used for process building and ETL transformations is Kettle. Using this tool, the ETL procedure it is reduced to creating one or more transformation projects, then saved to files.

A transformation begins with the retrieval of data by one or multiple source database tables. In this case the only source is the database reconciled in Figure , after which data processing operations are performed and then ends with the update of the target Date Warehouse.

Figure 6.24 shows the transformation that populates the Data Warehouse

A transformation, to recreate the cube related to a fact, begins with the table input step that enables the data extraction from the data sources. The only source from which data is extracted is the reconciled database, which is shown in the ER diagram in Figure .

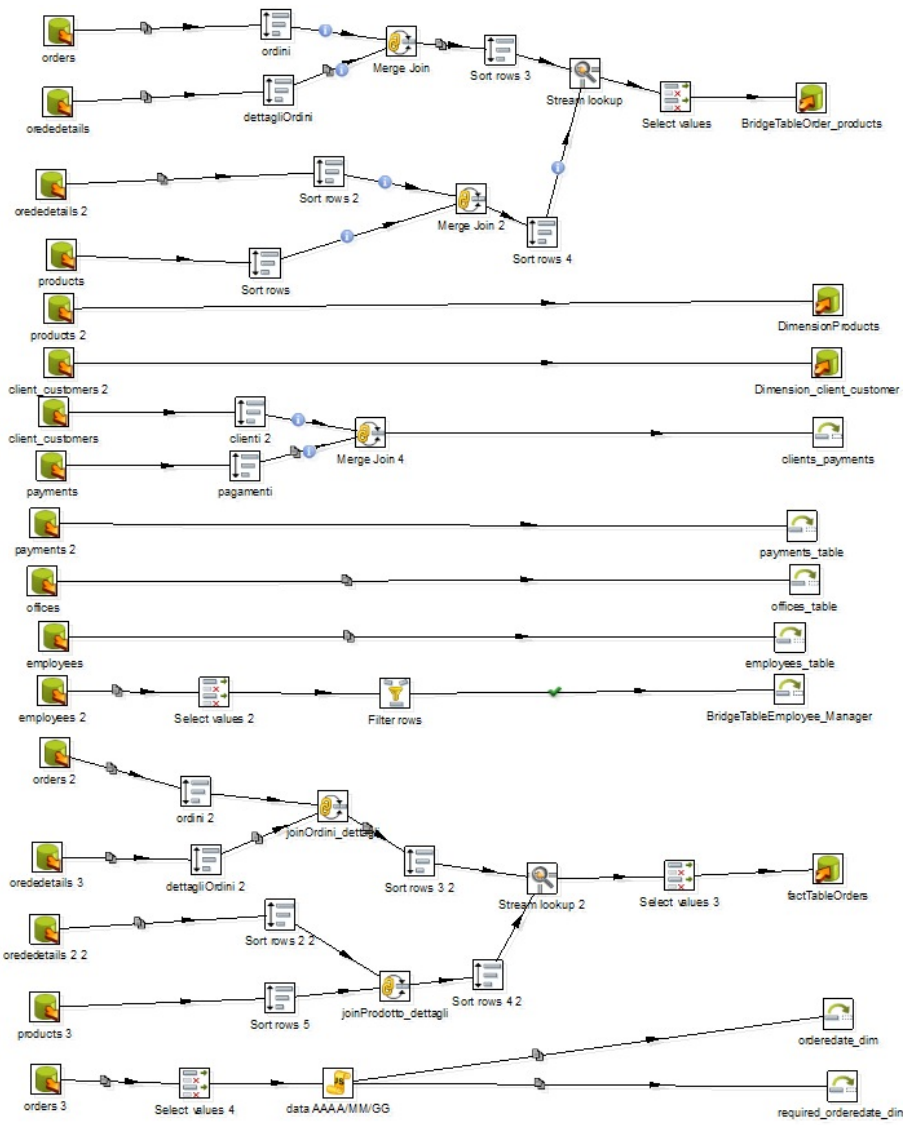


Figure 6.24. Example of ETL transformation that populates the Data Warehouse

Subsequently, during data extraction, if the source table is the same as the destination table, i.e., there is a direct mapping between the two, the destination table of the Data Warehouse is directly populated. Otherwise the *<merge join>* step can be used to make appropriate joins between the source tables. The *<merge join>* step allows to make both the inner join and the full outer join (left and right) of the two input tables by providing the primary key for each one table. It is often necessary to use the *<Select values>* step to select some fields from the source input tables to perform ad hoc modeling on the values of the selected fields. In particular, using the *<Modified Java Script Value>* step, the value of an input field is changed, using the JavaScript functions provided by Kettle. The functions made available are divided by the type of input parameters. For example, there are string, numeric, data, logic, special and file functions. Using the *<Filter rows>* step you select those values, and therefore those tuples that respect a given condition, and they will be routed to the next steps of the transformation according to the result that discriminates them. Once all the steps have been completed, the tables of the Data Warehouse are populated using the *<Insert Update>* step or using the *<Table Output>* step provided for inserting or updating the tables with the input tuples based on their primary keys.

6.5 OLAP analysis of the cube and strategic reports

The last phase of the proposed process is the analysis of the generated OLAP scheme. The OLAP analysis is based on two components, Mondrian, for the management of the multidimensional cube and for the retrieval of the data, JPivot, for their visualization.

6.5.1 Multi-dimensional scheme queries

At this point the cube is ready to be interrogated. MDX is the main language used in Mondrian for the formulation of queries on a cube. In Figure 6.25, the OLAP table relative to the cube *orders_cube*, the result of the default MDX query used to initialize the table, is displayed by JPivot. On the lines all the dimensions are shown, while on the columns some of the numerical measures are identified.

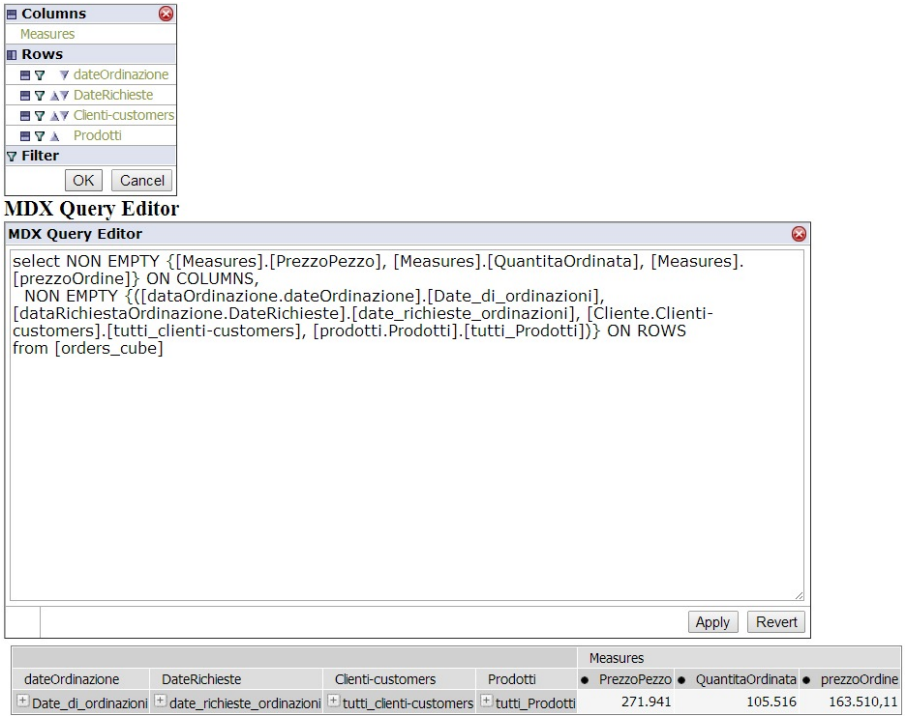


Figure 6.25. Visualization of the *orders_cube* cube schema

Figures 6.25 6.26 and 6.27 shows the result of the strategic queries.

Some useful graphs are also used for strategic analysis.

For example, Figures 6.28, 6.29, 6.30 and 6.31 shows the sales trend between 2003 and 2005.

(All)	cliente	Pagamenti	Impiegato	Ufficio	Manager	PrezzoPezzo	QuantitaOrdinata	prezzoOrdine
⊟ tutti_clienti-customers						271.941	105.516	163.510,11
tutti_clienti-customers	⊟ 103					1.806	810	983,67
	103	⊟ HQ336336				602	270	327,89
		HQ336336	⊟ 1102			602	270	327,89
			1102	⊟ 4		602	270	327,89
				4	1056	602	270	327,89
		⊞ JM555205				602	270	327,89
		⊞ OM314933				602	270	327,89
	⊟ 114					20.644	7.704	12.596,84
	114	⊟ GG31455				5.161	1.926	3.149,21
		GG31455	⊟ 1088			5.161	1.926	3.149,21
			1088	⊟ 6		5.161	1.926	3.149,21
				6	1056	5.161	1.926	3.149,21
		⊞ MA765515				5.161	1.926	3.149,21
		⊞ NP603840				5.161	1.926	3.149,21
		⊞ NR27552				5.161	1.926	3.149,21
	⊟ 119					13.665	5.496	8.479,08
	119	⊟ DB933704				4.555	1.832	2.826,36
		DB933704	⊟ 1337			4.555	1.832	2.826,36
			1337	⊟ 4		4.555	1.832	2.826,36
				4	1102	4.555	1.832	2.826,36
		⊞ LN373447				4.555	1.832	2.826,36
		⊞ NG94694				4.555	1.832	2.826,36
	⊟ 121					12.432	4.328	7.395,48
	121	⊟ DB889831				3.108	1.082	1.848,87
		DB889831	⊟ 1501			3.108	1.082	1.848,87
			1501	⊟ 7		3.108	1.082	1.848,87
				7	1102	3.108	1.082	1.848,87
		⊞ FD317790				3.108	1.082	1.848,87
		⊞ KI831359				3.108	1.082	1.848,87
		⊞ MA302151				3.108	1.082	1.848,87
	⊟ 124					150.696	57.294	89.992,17

Figure 6.26. Managed orders for each business area query

Prodotti	Clienti-customers				Measures		
prodotto	(All)	cliente	Pagamenti	Impiegato	PrezzoPezzo	QuantitaOrdinata	prezzoOrdine
S10_2016	⊟tutti_clienti-customers				27.994	10.878	16.364,39
	tutti_clienti-customers	⊟103			501	213	279,66
		103	⊟HQ336336		167	71	93,22
			HQ336336	⊞1102	167	71	93,22
			⊞JM555205		167	71	93,22
			⊞OM314933		167	71	93,22
		⊟114			10.516	4.088	6.280,6
		114	⊟GG31455		2.629	1.022	1.570,15
			GG31455	⊞1088	2.629	1.022	1.570,15
			⊞MA765515		2.629	1.022	1.570,15
			⊞NP603840		2.629	1.022	1.570,15
			⊞NR27552		2.629	1.022	1.570,15
		⊟119			4.362	1.803	2.728,47
		119	⊟DB933704		1.454	601	909,49
			DB933704	⊞1337	1.454	601	909,49
			⊞LN373447		1.454	601	909,49
			⊞NG94694		1.454	601	909,49
		⊟129			2.178	642	1.192,5
		129	⊟DM826140		726	214	397,5
			DM826140	⊞1165	726	214	397,5
			⊞ID449593		726	214	397,5
			⊞PI42991		726	214	397,5
		⊟131			6.711	2.283	3.977,13
		131	⊟CL442705		2.237	761	1.325,71
			CL442705	⊞1323	2.237	761	1.325,71
			⊞MA724562		2.237	761	1.325,71
			⊞NB445135		2.237	761	1.325,71
		⊟141			8.736	3.367	4.682,34
		141	⊟AU364101		672	259	360,18
			AU364101	⊞1401	672	259	360,18
			⊞DB583216		672	259	360,18
			⊞DL460618		672	259	360,18
			⊞HJ32686		672	259	360,18

Figure 6.27. Third business rule report

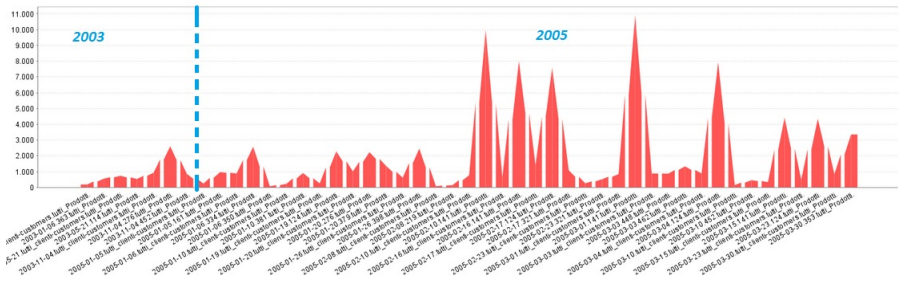


Figure 6.28. Analysis of the sales trend between 2003 and 2005

dateOrdinazione	Clienti-customers		Measures		
DataOrdine	(All)	cliente	PrezzoPezzo	LineaPodotto	prezzoOrdine
2003-01-06	⊟tutti_clienti-customers		301	Vintage Cars	185,01
2003-01-06	tutti_clienti-customers	⊞363	903	Vintage Cars	555,03
2003-05-21	⊟tutti_clienti-customers		1.203	Vintage Cars	735,99
2003-05-21	tutti_clienti-customers	⊞114	908	Vintage Cars	532,72
2003-11-04	⊟tutti_clienti-customers		1.542	Motorcycles	937,49
2003-11-04	tutti_clienti-customers	⊞276	4.528	Motorcycles	2.605,6
2003-11-04	tutti_clienti-customers	⊞452	1.230	Classic Cars	858,27

Slicer: [(All)=tutti_Prodotti]

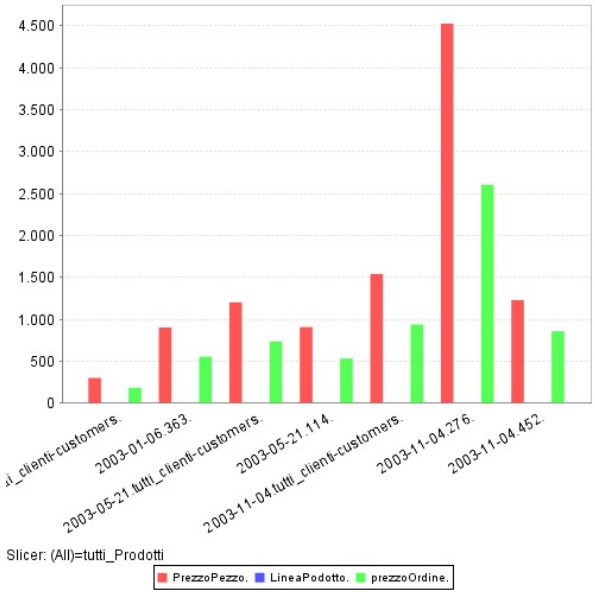


Figure 6.29. Year 2003 orders

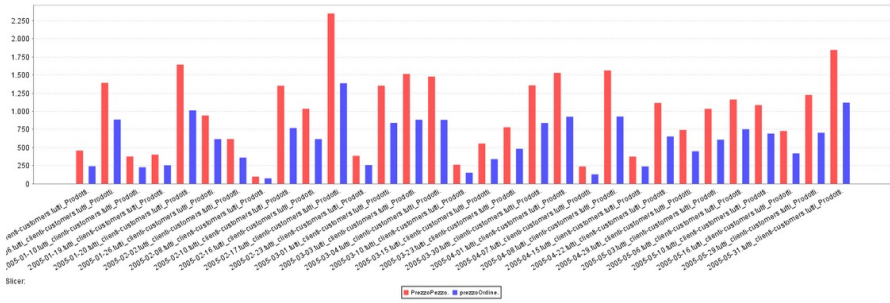


Figure 6.30. Year 2005 orders

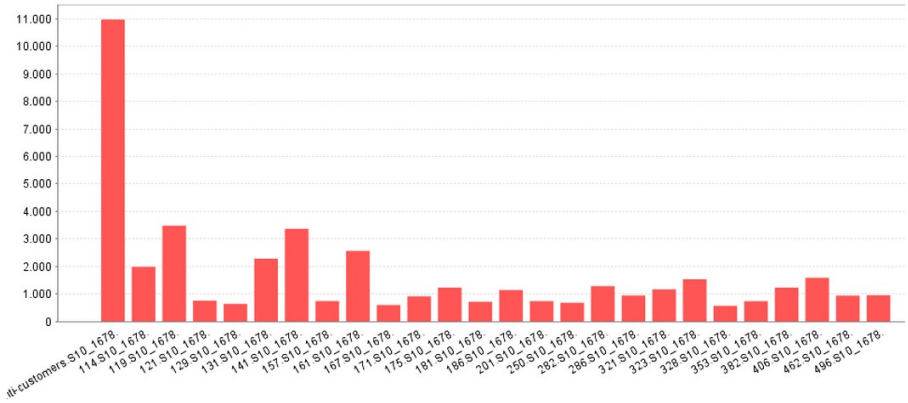


Figure 6.31. Customer's costs during the last three years

Chapter 7

Case studies

This chapter presents a case study by applying the process described in this thesis on three different types of data source. The goal is to assess the validity of the proposed process and highlight possible limitations.

For each example, only the first phase of the process will be illustrated, ie.e, up to the source integration phase, because the second phase is strongly dependent on the previous one and it is less problematic as it follows the classic approach to building a scheme OLAP [61].

The purpose of this Chapter is therefore to validate the data integration phase, which is the most critical phase of the entire presented process.

In the following, the details of the three data sources used for validation are listed:

- PANDA. It consists of two relational databases whose tables sources are populated by functions that automatically compose the values of each record; in other words the tuples are self-generated. The particularity of this example is that data contained in the sources is enormous;
- VIVAIO. It is made up of two relational databases whose source tables show

a diagram constructed in such a way as to promote integration. In this case, the amount of data contained in the sources is reduced;

- RECIPES. It consists of three CSV files, with real schemas and tuples.

To validate the results obtained, our oracle is represented by the Data Warehouse obtained by hand-made experts work. The parameters of configuration, such as the error thresholds to be set at each stage, the choice of k for executing the k-means algorithm and the coefficients for the comparison function of the conducted analyzes, are reported specifically for each case study.

7.1 First case study: PANDA

The purpose of the first case study is to define a PANDA (Presences And Notes Data Analysis using Data Warehousing) information system, containing a Data Warehouse that allows performing analysis on typical university activities, focusing the attention on the productivity of students. Data sources are represented by two relational databases, in particular:

- S.D.I. (Student Digital Identification) is a system for detecting university courses attendance, used between 2008 and 2010 at the University of Molise. Currently, the system database contains about 18,000 tuples related to attendance and over 54,000 related to the users.

- The UnderDesk system is a web application that allows users to be able to share notes. The system is currently in use and retains over 54,000 users and 3,000 notes.

Figure 7.1 shows the ER diagram of each source of data, the S.D.I. system is on the left hand side (red rectangle), while the underdesk system is on the right

hand side (green rectangle). Tables are extracted using the ETL Kettle tool and then exported as CSV file.

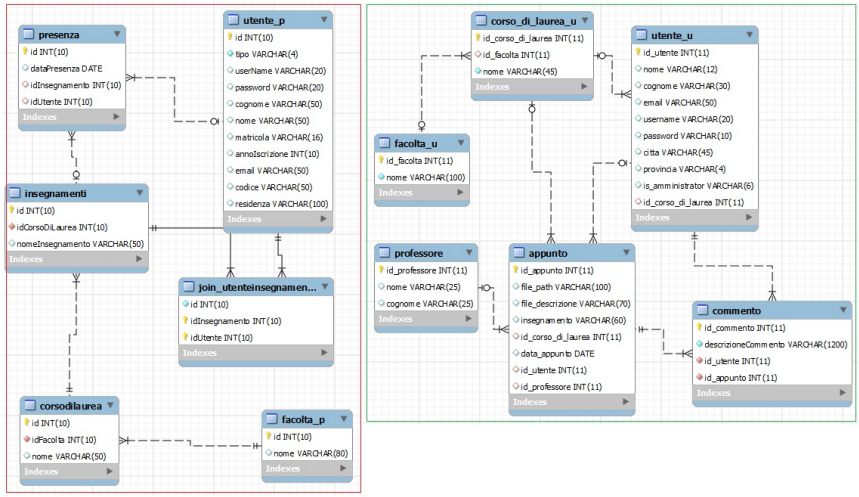


Figure 7.1. E/R diagrams of PANDA case study data sources

7.1.1 Data preprocessing

This phase aims to clear the data of the two sources, then the document matrix is constructed. The term-document matrix dimension is 95330 terms by 9 documents.

The comment table is deleted as not useful for the purpose of the information system to be constructed.

Similarly, the presence tables as well as the *join_utenteinsegnamento* table are deleted because they only contain numeric attributes. In cases of homonymy, the the rules described in Chapter 6 are applied. As a result, the tables of the system S.D.I. are named adding the suffix $\_p$ to the name of the table, while the UnderDesk system files are named by adding the suffix $\_u$ to the name of the tables.

7.1.2 Syntactic and semantic analysis

During the syntactic analysis the Levenshtein distances matrix is calculated and the cluster analysis is performed. A first analysis suggests the existence of two well sparated clusters, $\{facolta\_p, facolta\_u\}$ and $\{corsodilaurea, corso\_di\_laurea\_u\}$.

There cluster analysis using the k-means method is summarized in Table 7.3. For simplicity, documents are referenced by the number assigned in following list:

1. appunto
2. corso_di_laurea_u
3. corsodilaurea
4. facolta_p
5. facolta_u
6. insegnamenti
7. professore
8. utente_p
9. utente_u

The k-means algorithm found C_2 and C_4 clusters, which morphologically, suggest some inaccuracy in the clustering. In fact, the semantics of the schemes suggests that C_4 cluster should contain the $user\_u$ and $user\_p$ tables as they are eligible for integration. More details are expected from the semantic analysis.

There is a substantial difference with the execution performed in the syntactic analysis of the same algorithm with the same parameters.

Table 7.1. PANDA case study: k-means execution for syntactic analysis

Iteration	Parameters		Distance matrix documents	Clusters	Between_SS/ total_SS(in%)	Documents to remove
	k	n				
1	5	9	1,2,3,4,5,6,7,8,9	C_1={1} C_2={2,3,4,5,6} C_3={7} C_4={8} C_5={9}	100	7
2	4	8	1,2,3,4,5,6,8,9	C_1={1} C_2={2,3,4,5,6} C_4={8} $C5$={9}	100	1
3	3	7	2,3,4,5,6,8,9	C_2={2,3,4,5,6} C_4={8} C_5={9}	100	9
4	2	6	2,3,4,5,6,8	C_2={2,3,4,5,6} C_4={8}	87.2	-

Table 7.2. PANDA case study: k-means execution for semantic analysis

Iteration	Parameters		Distance matrix documents	Clusters	Between_SS/ total_SS(in%)	Documents to remove
	k	n				
1	5	9	1,2,3,4,5,6,7,8,9	C1={1} C2={2,3,4,5,6} C3={7} C4={8} C5={9}	96.6	7
2	4	8	1,2,3,4,5,6,8,9	C1={1} C2={2,3,4,5,6} C4={8} C5={9}	94.0	6
3	3	7	2,3,4,5,6,8,9	C2={2,3,4,5,6} C4={8} C5={9}	92.4	1
4	2	6	2,3,4,5,6,8	C2={2,3,4,5,6} C4={8}	89.9	-

7.1.3 Comparison of analysis results

. As noted, the results of the two analyzes are different, intersecting them there is only one cluster that reports similarities between elements, the cluster C2.

By analyzing the match matrix, note that key lists for the merges are the following for the tables below:

- utente_p = [usernameSDI, surnameSDI, nameSDI, emailSDI]
- user_u = [usernameU, usernameU, nameU, emailU]

7.1.4 Source Integration

No further cleansing phase is needed, so we can proceed with the integration by performing the internal joining of the two tables. The resulting new table has 16 columns, of which 4 are in common with the two tables (note that each original table has 10 columns).

7.2 Second case study: Vivaio

The purpose of the second case study is to build a Data Warehouse for the management of nursery services, integrating two sources of data from two different companies working in the nursery sector.

The two databases are:

- *Vivaio E-commerce* is the database of a nursery company that sells plants and provides related services;
- *Vivaio Interventi* is the database of a company that deals with it exclusively for the provision of nursery services, consisting of activities similar to those carried out by the first company.

Figures 7.2 and 7.3 show the diagrams of the two database. The red box contains the ER of the first database (Vivaio E-commerce), while the green box contains the ER of the second database (Vivaio Interventi).

Even in this case, tables are extracted using the ETL Kettle tool and exported then as CSV file.

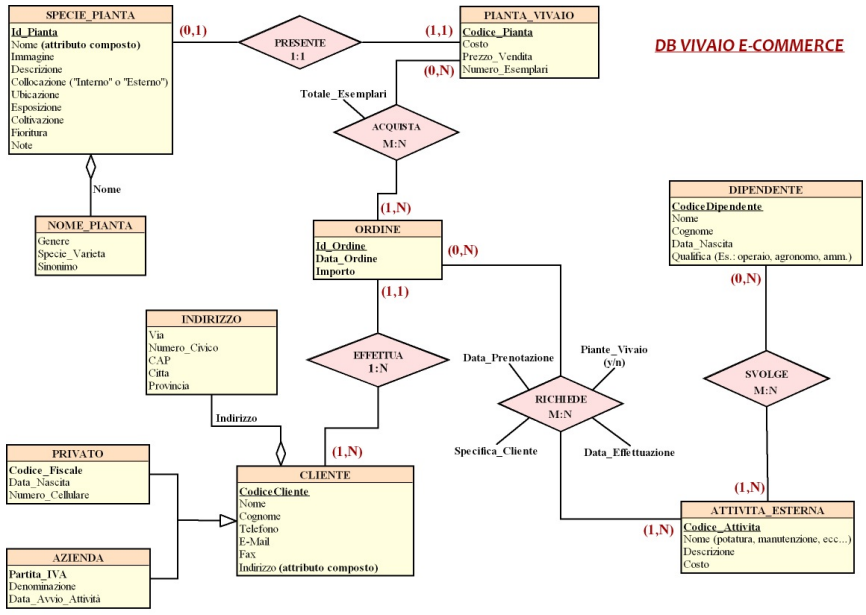


Figure 7.2. E/R diagram of the Vivaio E-Commerce database

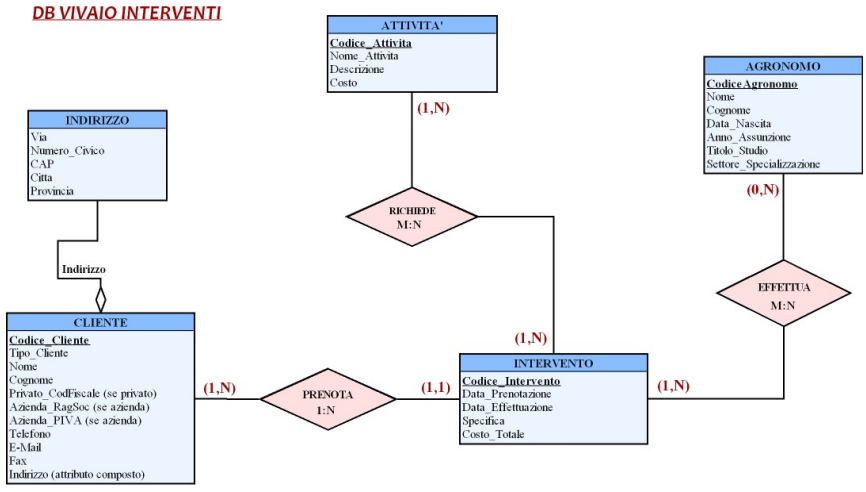


Figure 7.3. E/R diagram of the Vivaio Interventi database

7.2.1 Data preprocessing

This phase aims to clear the data of the two sources, then the term-document matrix is constructed, which in this case has 1279 terms and 9 documents. Columns for tables having only numeric attributes are removed.

In cases of homonymy, the the rules described in Chapter 6 are applied. As a result, database name has been added as suffix when needed. The resulting table names are:

- agronomo
- attivita
- attivitaesterna
- azienda
- cliente_interventi
- cliente_vivaio
- dipendente
- privato
- specie_pianta

7.2.2 Syntactic and semantic analysis

During the syntactic analysis the Levenshtein distances matrix is calculated and the cluster analysis is performed.

From the execution of the k-means it can be noticed that the *cliente* documents is lost in both databases, thus suggesting a possible correlation.

It can be noted a substantial difference with the execution of the same algorithm, with the same parameters, carried out in syntactic analysis and again the

Table 7.3. Vivaio case study: k-means execution for syntactic analysis

Iteration	Parameters		Distance matrix documents	Clusters	Between_SS/ total_SS(in%)	Documents to remove
	k	n				
1	5	9	1,2,3,4,5,6,7,8,9	C1={1} C2={2,3,4,5,6} C3={7} C4={8} C5={9}	96.6	7
2	4	8	1,2,3,4,5,6,8,9	C1={1} C2={2,3,4,5,6} C4={8} C5={9}	94.0	6
3	3	7	2,3,4,5,6,8,9	C2={2,3,4,5,6} C4={8} C5={9}	92.4	1
4	2	6	2,3,4,5,6,8	C2={2,3,4,5,6} C4={8}	89.9	-

Table 7.4. Vivaio case study: k-means execution for semantic analysis

Iteration	Parameters		Distance matrix documents	Clusters	Between_SS/ total_SS(in%)	Documents to remove
	k	n				
1	5	9	1,2,3,4,5,6,7,8,9	C1={1} C2={2,3} C3={4,7,8} C4={9} C5={5,6}	92.5	9
2	4	8	1,2,3,4,5,6,8,9	C1={1} C2={2,3} C3={4,7,8} C5={5,6}	91.5	1
3	3	7	2,3,4,5,6,8,9	C2={2,3} C3={4,7,8} C5={5,6}	92.4	-

C5 cluster that syntactic analysis fails to locate has been found in the semantic analysis.

The integration is performed by applying the inner joining to the two tables. The resulting table consists of 18 columns (8 are in common with the two original tables).

7.3 Third case Study: Ricette

As a last case study, we integrate data sources in the Italian kitchen context. In particular, the data sources are three CSV files that can be downloaded for free from the online open data catalog [1]

In particular, they include:

- Traditional Trentino products: a list of traditional Trentino Alto Adige's products by name, category, curiosity, description, conservation / processing methods and production area;
- Typical Trentino recipes: a collection of typical Trentino Alto Adige's recipes with traditional products (appetizers, first courses, main courses, desserts);
- Where to eat: containing essential data relating to easing in Emilia Romagna.

The aim is to test the integration process on real data, and then voluntarily we have chosen an example that shows the validity of the process. In fact the first two datasets are related, while the third is not related to any of the first two.

We expect only the first two to be integrated.

Here are the source schemas:

- recipes = [RecipeID, Title, Category, Ingredient, Preparation]
- productsTraditional = [category, url, syntetic product description, curiosity, methods of processing and storage]
- hospitality = [datasource, title, description, urlscheda, url image, description, environment, families, backgrounds, ISTAT code, province, address, location, latitude, longitude]

[1] These are open data made available by the Italian Ministry of Public Administration and downloadable at http://www.dati.gov.it/catalog

7.3.1 Data preprocessing

This phase aims to clear the data of the two sources, then the term-document matrix is constructed, resulting in 22598 terms and 3 documents.

7.3.2 Syntactic and semantic analysis

Figures 7.5 and 7.6 present the result of the execution of k-means algorithm for syntactic and semantic analysi, respectively.

Table 7.5. Ricette case study: k-means execution for syntactic analysis

Iteration	Parameters		Distance matrix documents	Clusters	Between_SS/ total_SS(in%)	Documents to remove
	k	n				
1	2	3	1,2,3	C1={1,2} C2={3}	94.5%	-

Table 7.6. Ricette case study: k-means execution for semantic analysis

Iteration	Parameters		Distance matrix documents	Clusters	Between_SS/ total_SS(in%)	Documents to remove
	k	n				
1	2	3	1,2,3	C1={1,2} C2={3}	99.6%	-

In this case study, the result of the k-means algorithm is identical for both syntactic analysis. Note the presence of the C1 cluster that shares the recipe and the traditional products documents. Then the correspondences matrix is calculated. Analyzing the the correspondences matrix we note that the key lists for the merge of the tables in question are respectively:

- recipes = [Title]
- traditionalProducts = [productname]

We proceed with the integration by performing the inner joining of the two tables. The new table obtained consists of 6 columns and 8 tuples, namely the traditional Trentino products, of which the recipe is also provided.

7.4 Results

Below we summarize the results obtained by executing the example shown in this Chapter and during the description of the process provided in Chapter 6.

Table 7.7 reports, for each case study, the following statistics:

- Name of the tested system;
- Peculiarities of the data sources, that is, whether the schemas and records were constructed in a proper way to encourage their integration;
- Number of sources;
- Number of source tables (and therefore of extracted CSV files) processed;
- Number of tuples of the integrated tables. Remember that each of the tests performs the integration of the less intuitive and more complex source pair, by convention the tables are named A and B;
- Accuracy. Is the record number of the reconciled scheme obtained by the process in the reconciled oracle schema, normalized by the total number of records in the reconciled oracle schema;
- Number of redundant columns. The superfluous columns maintained in the integrated scheme. An example is the column Address, which does not respect the first normal form and that is the concatenation of the columns Street, City, State.

Summarizing, In summary, it can be established that the process is valid in the 17 to 100% range of cases. In particular, data-level testing was valid for the three case studies. The case of nursery companies considerably reduces the validity of the process since it presents inconsistencies at the level of data in the sources, whose integration gives rise to a reconciled table containing only one tuple. This

Table 7.7. Case studies statistics

Case study	Peculiarities	# of sources	# of source tables	# of integrated table tuples	Accuracy	# of redundant columns
Car shopping	Ad-hoc schema real records	2	8	A=122 B=122	100%	1
PANDA	Auto-generated data values	2	9	A=54458 B=54110	99%	1
Vivaio	Ad-hoc schema real records	2	9	A=32 B=14	100%	0
Ricette	Real schema real records	2	3	A=49 B=14		0

result is not caused by an error in the process but is generated from the source data themselves.

Column redundancy is a disadvantage, but does not affect the results obtained because the "useful" columns of the integrated table are determined correctly and the constraints of referential integrity are maintained allowing for an easy implementation of the integrated scheme.

However the integrated table requires memory and is the subject of more expensive OLAP processing in terms of execution time.

In conclusion, the defined process uses techniques whose advantages are well documented. However, errors are possible because the Data Mining and Machine Learning techniques work by looking at the intrinsic dimension of the input data, which in this case are the populated tables of the sources. So a minimum loss of information is expected and this can lead to errors. In this process the errors are accepted within a certain threshold imposed a priori by the manager.

This limitation is the reason why the processi is semi-automatic. In fact, even if no manager's competence is required, he is requested to determine the error thresholds in an appropriate manner to avoid trivial or misleading results that can undermine strategic decisions.

Another disadvantage that involves the use of this process is the construction of a single OLAP scheme when it is necessary to model more than one concept of interest.

Finally, the definition of measures becomes very time-consuming to navigate the existing relationships, moreover the (open source) tools used for the implementation of the process, such as Mondrian and JPivot, can constrain the modeling, as well as the visualization of the data.

Chapter 8

Conclusion

Nowadays, strategic information is one of the main assets for many organizations and, in the next future, it will become increasingly more important for their health and survival [1].

Data Warehouse Systems were born to store huge amount of heterogeneous data in order to be successively analyzed and extracted according to the necessity of a manager that needs to take strategic decisions and plan the evolution and the direction of his organization.

For a proper decision-making, a DWS has to contain a vast amount of heterogeneous data coming from several organizational operational systems and external sources [2]. However, it is very difficult and onerous to integrate several data from independent databases containing semantic differences and no specific or canonical concept description [3].

Moreover, the costs to develop a DW are usually high and the results are often unpredictable. Furthermore, it frequently happens that both customers and developers are dissatisfied with the results of a DW project since these projects often are characterized by a long delay in system delivering and by a large number of missing or inadequate requirements [4].

Dashboards are essential tools for monitoring business performance and empowering managers to make better decisions faster. The challenge, then, is to select the best data visualization for a given business.

This research was initiated by proposing a tool and a graphical approach for extracting data from a data warehouse and representing them in a complex graphical form providing information on the relationships among its graph components. There was the need of deeply understanding whether this approach effectively improves operational efficiency and decision making.

In this thesis, we assessed the data visualization based on complex graphs by conducting a controlled experiment. The aim of this investigation was to determine whether this approach is effective when users analyze real data. Participants were asked to comprehend the meaning of data using traditional Dashboard diagrams and complex graph diagrams. The effort required to answer an evaluation questionnaire has been assessed together with the comprehension of the data representations and the results have been presented and discussed

The results of the evaluation suggest that people reached a higher comprehension when using Complex Graphs like the ones produced by our approach as compared to standard Dashboard Graphs. Furthermore, the results related to the comprehension time indicated that the participants spent significantly less time to understand a problem adopting a Complex Graphs visualization as compared to standard Dashboard Graphs. Both high and low ability users took advantages from the complex graph representation in term of both comprehension and effort, with high ability participants taking a greater advantage in comprehension time. This finding is relevant for the decision maker: complex graphs represent an effective visualization approach that enables a quicker and higher comprehension of data, improving the appropriateness of the decisions and reducing the decision making effort.

Thus, we can conclude that this kind of graphs can be adopted for making

decisions since it reduces interpretation effort and increase data interpretation. In addition, the user perception questionnaire revealed that participants expressed a positive judgment on the complex graphs. The experiment objects were realistic enough for manager decisions. However, additional investigations should be conducted to further verify that the achieved results scale to executive decisions. This point is of interest for the future work.

Based on this finding, in Chapter 5 an automatic generation of data warehouse has been presented that uses complex graphs to visualize the main facts that a decision maker should use for strategic decisions. In particular, the CoDe language has been adopter to represent such information.

When assessing the automatic generation of data warehouse, we noticed that the most critical phase of the process was the data integration. In fact, this phase needs of a major contribution from the manager. So, to encourage the implementation of the process in a real setting, a semi-automatic data integration process has been proposed in Chapter 6.

Finally, the proposed process has been tested on 4 case studies to assess the goodness of the approach. The results, indicate that, despite some limitations, such as the need to ask the user to specify the error threshold and some inefficiencies in terms of memory usage and computation time, e.g., in the case of column redundancies, in three of the four case studies we obtained encouraging results.

Directions for future works on this research should certainly include the evaluation of the overall process in a real setting, working side-by-side with decision makers to define a more efficient language to describe strategic goals. In fact, the natural language is often the main cause of a bad behavior of similar approaches due to vague or ambiguous specifications.

Moreover, new representations of complex diagrams should be experimented to understand if the views used in this thesis can be further improved.

Finally, an integrated platform that guides the user though the entire process,

allowing visual interaction mechanisms to provide input and feedbacks to the process, as well as to interactively browse complex diagrams would certainly help non-expert decision makers during the enactment of the proposed process.

Bibliography

[1] P. Ponniah, *Data Warehousing Fundamentals*, 1st ed. New York, NY, USA: John Wiley & Sons, Inc., 2001.

[2] S. Saroop and M. Kumar, "Comparison of data warehouse design approaches from user requirement to conceptual model: A survey," in *Proceedings of the 2011 International Conference on Communication Systems and Network Technologies*, ser. CSNT '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 308–312. [Online]. Available: http://dx.doi.org/10.1109/CSNT.2011.161

[3] S. Castano, V. De Antonellis, and S. De Capitani di Vimercati, "Global viewing of heterogeneous data sources," *IEEE Trans. on Knowl. and Data Eng.*, vol. 13, no. 2, pp. 277–297, Mar. 2001. [Online]. Available: http://dx.doi.org/10.1109/69.917566

[4] M. Golfarelli, S. Rizzi, and E. Turricchia, "Modern software engineering methodologies meet data warehouse design: 4wd," in *Data Warehousing and Knowledge Discovery*, ser. Lecture Notes in Computer Science, A. Cuzzocrea and U. Dayal, Eds. Springer Berlin / Heidelberg, 2011, vol. 6862, pp. 66–79.

[5] S. R. Gardner, "Building the data warehouse," *Commun. ACM*, vol. 41, pp. 52–60, September 1998. [Online]. Available: http://doi.acm.org/10.1145/285070.285080

[6] E. Soler, V. Stefanov, J.-N. Mazon, J. Trujillo, E. Fernandez-Madina, and M. Piattini, "Towards comprehensive requirement analysis for data warehouses: Considering security requirements," in *Proceedings of the 2008 Third International Conference on Availability, Reliability and Security*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 104–111. [Online]. Available: http://dl.acm.org/citation.cfm?id=1371602.1371880

[7] R. J. Santos and J. Bernardino, "Real-time data warehouse loading methodology," in *Proceedings of the 2008 international symposium on Database engineering & applications*, ser. IDEAS '08. New York, NY, USA: ACM, 2008, pp. 49–58. [Online]. Available: http://doi.acm.org/10.1145/1451940.1451949

[8] L. Chen, J. W. Rahayu, and D. Taniar, "Towards near real-time data warehousing." in *AINA'10*, 2010, pp. 1150–1157.

[9] G. Thakur and A. Gosain, "Dwevolve: a requirement based framework for data warehouse evolution," *SIGSOFT Softw. Eng. Notes*, vol. 36, pp. 1–8, Nov. 2011. [Online]. Available: http://doi.acm.org/10.1145/2047414.2047433

[10] M. Risi, M. Sessa, M. Tucci, and G. Tortora, "Code modeling of graph composition for data warehouse report visualization," *Knowledge and Data Engineering, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2013.

[11] S. Rizzi, A. Abelló, J. Lechtenbörger, and J. Trujillo, "Research in data warehouse modeling and design: dead or alive?" in *Proceedings of the 9th ACM international workshop on Data warehousing and OLAP*, ser. DOLAP '06. New York, NY, USA: ACM, 2006, pp. 3–10. [Online]. Available: http://doi.acm.org/10.1145/1183512.1183515

[12] J. Trujillo and S. Lujan-Mora, "A uml based approach for modeling etl processes in data warehouses," in *Conceptual Modeling - ER 2003*, ser. Lecture Notes in Computer Science, I.-Y. Song, S. Liddle, T.-W. Ling, and P. Scheuermann, Eds. Springer Berlin / Heidelberg, 2003, vol. 2813, pp. 307–320.

[13] J. Pardillo, M. Golfarelli, S. Rizzi, and J. Trujillo, "Visual modelling of data warehousing flows with uml profiles," in *Data Warehousing and Knowledge Discovery*, ser. Lecture Notes in Computer Science, T. Pedersen, M. Mohania, and A. Tjoa, Eds. Springer Berlin / Heidelberg, 2009, vol. 5691, pp. 36–47.

[14] P. Vassiliadis, A. Simitsis, and E. Baikousi, "A taxonomy of etl activities," in *Proceedings of the ACM twelfth international workshop on Data warehousing and OLAP*, ser. DOLAP '09. New York, NY, USA: ACM, 2009, pp. 25–32. [Online]. Available: http://doi.acm.org/10.1145/1651291.1651297

[15] A. Simitsis, P. Vassiliadis, and T. Sellis, *Optimizing ETL Processes in Data Warehouses*. Ieee, 2005, vol. 2, no. Icde, pp. 564–575. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1410172

[16] F. I. Anfurrutia, O. Diaz, and S. Trujillo, "A product-line approach to database reporting." in *JISBD'05*, 2005, pp. 163–170.

[17] K. C. Hsu and M.-Z. Li, "Applying clustering analysis on grouping similar olap reports," in *Proceedings of the 2010 Second International Conference on Computer Engineering and Applications - Volume 02*, ser. ICCEA '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 417–423. [Online]. Available: http://dx.doi.org/10.1109/ICCEA.2010.231

[18] H. Su and J. Su, "A study and practice of report system techniques based on three-layer calculating architecture," *Education Technology and Computer Science, International Workshop on*, vol. 2, pp. 654–657, 2010.

[19] M. Kulkarni, M. Lu, and D. Zhang, "A case-based data warehousing courseware," in *Proceedings of the IEEE International Conference on Information Reuse and Integration, IRI 2010, 4-6 August 2010, Las Vegas, Nevada, USA*. IEEE Systems, Man, and Cybernetics Society, 2010, pp. 245–248.

[20] D. Habich, W. Lehner, and M. Just, "Materialized views in the presence of reporting functions," in *Proceedings of the 18th International Conference on Scientific and Statistical Database Management*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 159–168. [Online]. Available: http://dl.acm.org/citation.cfm?id=1154779.1154995

[21] W. Lehner, W. Hummer, and L. Schlesinger, "Processing reporting function views in a data warehouse environment." in *ICDE'02*, 2002, pp. 176–176.

[22] H. Du and Y. Li, "The design and realization of the universal report system in the power system based on physical view," in *Proceedings of the 2010 International Conference on Web Information Systems and Mining - Volume 01*, ser. WISM '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 404–408. [Online]. Available: http://dx.doi.org/10.1109/WISM.2010.18

[23] P. Vassiliadis, "Arktos: towards the modeling, design, control and execution of etl processes," *Information Systems Journal*, vol. 26, no. 8, pp. 537–561, 2001. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0306437901000394

[24] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. Pearson Education, 2003. [Online]. Available: http://portal.acm.org/citation.cfm?id=773294

[25] T. Wojciechowski, B. Sakowicz, D. Makowski, and A. Napieralski, "Transaction system with reporting capability in a web-based data warehouse application developed in oracle application express," pp. 273–276, Feb 2009.

[26] E. Levy, J. Zacks, B. Tversky, and D. J. Schiano, "Gratuitous graphics? putting preferences in perspective." in *CHI*, B. A. Nardi, G. C. van der Veer, and M. J. Tauber, Eds. ACM, 1996, pp. 42–49.

[27] M. Schonlau and E. Peters, "Graph comprehension: An experiment in displaying data as bar charts, pie charts and tables with and without the gratuitous 3rd dimension," in *RAND Working Paper Series No. WR- 618*, 2008. [Online]. Available: http://ssrn.com/abstract=1272627

[28] O. Inbar, N. Tractinsky, and J. Meyer, "Minimalism in information visualization: Attitudes towards maximizing the data-ink ratio," in *Proceedings of the 14th European Conference on Cognitive Ergonomics: Invent! Explore!*, ser. ECCE '07. New York, NY, USA: ACM, 2007, pp. 185–188.

[29] S. Bateman, R. L. Mandryk, C. Gutwin, A. Genest, D. McDine, and C. Brooks, "Useful junk?: The effects of visual embellishment on comprehension and memorability of charts," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '10. New York, NY, USA: ACM, 2010, pp. 2573–2582. [Online]. Available: http://doi.acm.org/10.1145/1753326.1753716

[30] A. Quispel and A. Maes, "Would you prefer pie or cupcakes? preferences for data visualization designs of professionals and laypeople in graphic design," *Journal of Visual Languages & Computing*, vol. 25, no. 2, pp. 107–116, 2014.

[31] J. L. Santos, S. Govaerts, K. Verbert, and E. Duval, "Goal-oriented visualizations of activity tracking: A case study with engineering students," in *Proceedings of the 2Nd International Conference on Learning Analytics and Knowledge*, ser. LAK '12. New York, NY, USA: ACM, 2012, pp. 143–152.

[32] A. Fish, B. Khazaei, and C. Roast, "User-comprehension of euler diagrams," *Journal of Visual Languages & Computing*, vol. 22, no. 5, pp. 340–354, 2011.

[33] J. Bertin, *Semiology of Graphics*. University of Wisconsin Press, 1983.

[34] M. Risi, M. Sessa, M. Tucci, and G. Tortora, "CoDe Modeling of Graph Composition for Data Warehouse Report Visualization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 3, pp. 563–576, 2014.

[35] N. Juristo and A. Moreno, *Basics of Software Engineering Experimentation*. Kluwer Academic Publishers, 2001.

[36] B. Kitchenham, S. Pfleeger, L. Pickard, P. Jones, D. Hoaglin, K. El Emam, and J. Rosenberg, "Preliminary Guidelines for Empirical Research in Software Engineering," *IEEE Trans. Soft. Eng.*, vol. 28, no. 8, pp. 721–734, 2002.

[37] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in software engineering: an introduction*. Norwell, MA, USA: Kluwer Academic Publishers, 2000.

[38] V. R. Basili and H. D. Rombach, "The TAME project: Towards improvement-oriented software environments," *IEEE Trans. Software Eng.*, vol. 14, no. 6, pp. 758–773, 1988.

[39] F. Ricca, M. Di Penta, M. Torchiano, P. Tonella, and M. Ceccato, "How Developers' Experience and Ability Influence Web Application Comprehension Tasks Supported by UML Stereotypes: A Series of Four Experiments," *IEEE Trans. Softw. Eng.*, vol. 36, no. 1, pp. 96–118, 2010.

[40] Oppenheim, *Questionnaire design, interviewing & attitude measurement*. Pinter, 1992.

[41] S. S. Shapiro and M. B. Wilk, "An analysis of variance test for normality," *Biometrika*, vol. 3, no. 52, 1965.

[42] R. Grissom and J. Kim, *Effect Sizes For Research: A Broad Practical Approach*. Taylor & Francis Group, 2005. [Online]. Available: http://books.google.it/books?id=o5D-Ens7WvUC

[43] V. B. Kampenes, T. Dybå, J. E. Hannay, and D. I. Sjøberg, "A systematic review of effect size in software engineering experiments," *Information and Software Technology*, vol. 49, no. 11–12, pp. 1073 – 1086, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0950584907000195

[44] P. Ellis, *The Essential Guide to Effect Sizes: Statistical Power, Meta-Analysis, and the Interpretation of Research Results*. Cambridge University Press, 2010. [Online]. Available: http://books.google.it/books?id=5obZnfK5pbsC

[45] J. Verelst, "The influence of the level of abstraction on the evolvability of conceptual models of information systems," in *Proc. of the International Symposium on Empirical Software Engineering*. IEEE CS Press, 2004, pp. 17–26.

[46] R. Kaplan and D. Norton, *The Balanced Scorecard: Translating Strategy Into Action*, ser. Harvard Business School Press. Harvard Business School Publishing India Pvt. Limited, 1996.

[47] S.-K. Chang, V. Deufemia, G. Polese, and M. Vacca, "A normalization framework for multimedia databases," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 12, pp. 1666–1679, 2007.

[48] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: An update," *SIGKDD Explor. Newsl.*, vol. 11, no. 1, pp. 10–18, Nov. 2009. [Online]. Available: http://doi.acm.org/10.1145/1656274.1656278

[49] H. D, "Performance comparison of apriori and fp-growth algorithms in generating association rules," *Proceedings of the European Computing Conference*, pp. 376–381, 2011.

[50] P. A. Flach and N. Lachiche, "Confirmation-guided discovery of first-order rules with tertius," *Machine Learning*, vol. 42, pp. 61–95, 2001.

[51] D. Hunyadi, "Performance comparison of apriori and fp-growth algorithms in generating association rules," in *Proceedings of the 5th European conference on European computing conference*, ser. ECC'11. Stevens Point, Wisconsin, USA: World Scientific and Engineering Academy and Society (WSEAS), 2011, pp. 376–381. [Online]. Available: http://dl.acm.org/citation.cfm?id=1991016.1991084

[52] H. Lijun, L. Linghua, L. Xiaoniu, and W. Degao, "Comparison and analysis of algorithms for association rules," *First International Workshop on Database Technology and Applications*, pp. 196–198, 2009.

[53] R. Elmasri and S. Navathe, *Fundamentals of Database Systems*, 6th ed. USA: Addison-Wesley Publishing Company, 2010.

[54] P. Ciuccarelli, M. I. Sessa, and M. Tucci, "Code : a graphic language for complex system visualization," *ITAIS 2010*, pp. 1–8, 2010.

[55] M. Casters, R. Bouman, and J. v. Dongen, *Pentaho Kettle Solutions: Building Open Source ETL Solutions with Pentaho Data Integration*. Wiley Publishing, 2010.

[56] D. B. L. L. S. Montrone S., Perchinunno P., "Le tecniche di integrazione di dati le tecniche di integrazione di dati per lo studio della povertà," 2011.

[57] S. V. C. G. V. J. A. M. Ortiz E., Babbar A., "Multi source data integration for aircraft healt management."

[58] G. B. Yang Y., Chen M., "An effective content-based schema matching algorithm," 2008.

[59] F.-X. H. Guo-Hui L., Xiao-Kun D., "A schema matching method based on partial functional dependencies," 2008.

[60] Z. Y. Bourennani F., Q. Pu K., "Visual integration tool for heterogeneous data type by unified vectorization," 2009.
[61] A. Petti, "Generazione automatica di schemi olap a partire da business rule."