

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO



**Implementação de um mecanismo de  
extracção e carregamento de dados  
para o ALERT® DATA WAREHOUSE  
a partir do ALERT® PRIVATE  
PRACTICE**

**David de Almeida Marques**

Relatório de Projecto  
Mestrado Integrado em Engenharia Informática

Orientador: Prof. Rui Camacho

Julho de 2008



# **Implementação de um mecanismo de extracção e carregamento de dados para o ALERT® DATA WAREHOUSE a partir do ALERT® PRIVATE PRACTICE**

**David de Almeida Marques**

Relatório de Projecto

Mestrado Integrado em Engenharia Informática

Aprovado em provas públicas pelo Júri:

Presidente: Jorge Manuel Gomes Barbosa (Professor)

---

Arguente: José Luis Oliveira (Professor)

Vogal: Rui Camacho (Professor)

31 de Julho de 2008



# Confidencial

Nos termos do protocolo de estágio e do acordo de confidencialidade celebrado com a ALERT Life Sciences Computing, S.A. ("ALERT"), o presente relatório é confidencial e poderá conter referências a invenções, know-how, desenhos, programas de computador, segredos comerciais, produtos, fórmulas, métodos, planos, especificações, projectos, dados ou obras abrangidos por direitos de propriedade industrial e/ou intelectual da ALERT. Este relatório só poderá ser utilizado para efeitos de investigação e de ensino. Qualquer outro tipo de utilização está sujeita a autorização prévia e por escrito da ALERT.

In accordance with the terms of the internship protocol and the confidentiality agreement executed with ALERT Life Sciences Computing, S.A. ("ALERT"), this report is confidential and may contain references to inventions, know-how, drawings, computer software, trade secrets, products, formulas, methods, plans, specifications, projects, data or works protected by ALERT's industrial and/or intellectual property rights. This report may be used solely for research and educational purposes. Any other kind of use requires prior written consent from ALERT.



# Resumo

Este trabalho surgiu da inexistência de um produto de *data warehousing* sobre o ALERT® PRIVATE PRACTICE, aplicação clínica que gere o fluxo de dados clínicos numa clínica privada. O novo produto, ADW PRIVATE PRACTICE (ADW PP), destina-se ao arquivo e análise de informação clínica e operacional, permitindo a realização de pesquisas, análises e relatórios complexos, no contexto de clínicas privadas. A informação do ADW e do ALERT provêm de duas bases de dados distintas mas que assentam no motor Oracle. A base de dados do ADW, um *data warehouse*, assenta no modelo dimensional para maximizar a eficiência de acesso aos dados. Esse acesso é feito através duma interface *web* que acede aos dados presentes no *data warehouse*.

Este projecto envolveu a definição e implementação dum processo que permitisse ter os dados carregados no *data warehouse*, num formato adequado para visualização. Uma análise ao contexto de negócio foi feita de forma a encontrar indicadores clínicos, financeiros e administrativos adequados a uma clínica privada e à sua gestão. O sistema fonte e de destino foram também analisados de modo a encontrar a informação certa que respondesse a esses indicadores. O modelo de dados do ADW já contém bastante informação relativa a processos de negócio clínicos, pelo que os novos desenvolvimentos podem-se dividir em três grupos. O primeiro implica a criação duma nova estrela de agendamentos de consultas. Foi criada uma nova tabela de factos e duas dimensões novas no *data warehouse* do ADW. O processo de extracção, transformação e carregamento de dados (ETL) foi implementado usando a ferramenta Oracle Data Integrator (ODI). O segundo, e mais complexo, envolveu a reformulação total da maior estrela do ADW, a das tarefas dos profissionais. A extracção dos dados foi dividida em duas fases. Metade das tarefas foram extraídas do ALERT por meio do ODI e outra metade do ADW por meio de código PL/SQL. O carregamento de dados para uma nova tabela de factos foi feito usando o ODI e cruzando informação com uma nova dimensão criada para o efeito, o tipo de tarefa. Tabelas de agregação foram criadas de forma a maximizar a eficiência de perguntas temporais por parte dos utilizadores. O terceiro grupo de desenvolvimento esteve na origem de criação de novos factos nas estrelas existentes, julgados necessários para o ADW PP, e do carregamento de dados em inglês, nomeadamente de dados geográficos.

A garantida de qualidade do processo e dos dados foi uma preocupação constante no projecto e foram usadas várias técnicas de testes. Invariantes, tabelas de erros, testes funcionais, unitários, e de performance foram usados no decorrer do projecto para validar a implementação. Ao nível de objectivos, todos foram atingidos de forma satisfatória e a reformulação da estrelas da tarefas ainda superou os objectivos iniciais, com um melhoramento da performance relativamente ao antigo processo.



# Abstract

This work arose from the lack of a data warehousing product for the ALERT ® PRIVATE PRACTICE, clinical software to manage the flow of clinical data in a private clinic. The new product, ADW PRIVATE PRACTICE (ADW PP), is aimed at archiving and analyzing clinical and operational information, allowing the realization of research, analysis and complex reports, in the context of private clinics. Information from ADW and ALERT® come from two different databases but both rely on Oracle technology. The ADW database, a data warehouse, is based on a dimensional model to maximize the efficiency of data access. Data query is done through a web interface that connect to the data warehouse.

This project was involved in defining and implementing the process that load and prepares the data in the *data warehouse*, in an adequate format for visualization. An analysis of the business context was done in order to find clinical, financial and administrative indicators, appropriate to a private clinic and its management. The source and destination system were also analyzed, to find the right information to respond to these indicators. The ADW data model already contains information concerning the clinical workflow, so new developments can be divided into three new groups. The first involves the creation of a new star of schedules for consultations. A new facts table and two new dimensions were created in the ADW data warehouse. The process of extraction, transformation and loading of data (ETL) was implemented using the tool Oracle Data Integrator (ODI). The second, more complex, involved the complete revision of the biggest star of the ADW, the tasks of professionals. The data extraction was divided into two phases. Half of the tasks were extracted from the ALERT through ODI and the other half from the ADW through PL/SQL coding. The data loading to a new tasks facts table was made using ODI and the crossing of information with a new dimension created for this purpose, the type of task. Tables of aggregation were created in order to maximize the efficiency of temporal questions from users. The third group of development has led to creation of new facts in existing stars, deemed necessary for the ADW PP, and loading of data in English, notably geographic data.

The guarantee of quality of the process and data was a constant concern in the project and various techniques were used for testing. Invariants, error tables, functional, unit, and performance tests were used during the project and guaranteed the fulfilment of goals. In terms of objectives, all were achieved and the recasting of the star of tasks surpassed the original goals, with an improved performance compared to the old process.



# Agradecimentos

Quero, por este meio, agradecer a um conjunto de pessoas que me ajudaram durante a realização deste projecto e que proporcionaram uma primeira experiência profissional única. Ao Sr. Eng<sup>o</sup> Hugo Vieira, responsável pelo projecto na ALERT, agradeço pelo seu acompanhamento. Ao Prof. Rui Camacho, responsável pelo projecto na Faculdade de Engenharia da Universidade do Porto, agradeço pela sua orientação e disponibilidade. A Dra. Ana Côrte-Real, responsável pela equipa em que estive integrado, agradeço pelo apoio, liderança e disponibilidade durante todo o projecto. Ao Bruno Carolo e Miguel Duarte, colegas de desenvolvimento, agradeço pelo apoio constante, pela fácil integração que me proporcionaram e pelo excelente ambiente de trabalho criado. Agradeço ainda o apoio de toda a equipa envolvida neste projecto, particularmente ao Abel Cunha e Marco Murta pelo apoio prestado sempre que foi necessário.

Deixo ainda expresso os meus agradecimentos a todos os que participaram e que de alguma forma contribuíram para a realização e sucesso deste projecto.

David de Almeida Marques



*Dedico este relatório à Sara*



# Conteúdo

|          |                                                    |           |
|----------|----------------------------------------------------|-----------|
| <b>1</b> | <b>Introdução</b>                                  | <b>1</b>  |
| 1.1      | Contexto/Enquadramento . . . . .                   | 1         |
| 1.2      | Projecto . . . . .                                 | 3         |
| 1.3      | Motivação e objectivos . . . . .                   | 4         |
| 1.4      | Estrutura da dissertação . . . . .                 | 4         |
| <b>2</b> | <b>Estado da Arte</b>                              | <b>7</b>  |
| 2.1      | Data Warehouse: conceitos gerais . . . . .         | 7         |
| 2.2      | Modelo de dados de um Data Warehouse . . . . .     | 9         |
| 2.3      | Construção de um Data Warehouse . . . . .          | 14        |
| 2.4      | Processo ETL . . . . .                             | 17        |
| 2.5      | Tecnologias . . . . .                              | 20        |
| 2.5.1    | Oracle . . . . .                                   | 20        |
| 2.5.2    | PL/SQL Developer . . . . .                         | 21        |
| 2.5.3    | Oracle Designer . . . . .                          | 21        |
| 2.5.4    | MicroStrategy . . . . .                            | 22        |
| 2.5.5    | Oracle Data Integrator . . . . .                   | 23        |
| 2.5.6    | SVN . . . . .                                      | 26        |
| <b>3</b> | <b>Análise do problema</b>                         | <b>27</b> |
| 3.1      | Contextualização . . . . .                         | 27        |
| 3.1.1    | Sistema fonte . . . . .                            | 28        |
| 3.1.2    | Sistema de destino . . . . .                       | 29        |
| 3.1.3    | Objectivos . . . . .                               | 32        |
| 3.2      | Situação actual do ADW . . . . .                   | 33        |
| 3.2.1    | Arquitectura do ADW . . . . .                      | 33        |
| 3.2.2    | Processo ETL do ADW . . . . .                      | 35        |
| 3.2.3    | Indicadores existentes . . . . .                   | 40        |
| 3.3      | Necessidades para o ADW PRIVATE PRACTICE . . . . . | 41        |
| 3.4      | Limitações . . . . .                               | 43        |
| 3.5      | Resumo ou conclusões . . . . .                     | 45        |
| <b>4</b> | <b>Modelo de dados do ADW PRIVATE PRACTICE</b>     | <b>47</b> |
| 4.1      | Estrela dos agendamentos . . . . .                 | 47        |
| 4.2      | Estrela das tarefas dos profissionais . . . . .    | 50        |
| 4.2.1    | Alternativa 1 . . . . .                            | 51        |
| 4.2.2    | Alternativa 2 . . . . .                            | 53        |

## CONTEÚDO

|          |                                                 |           |
|----------|-------------------------------------------------|-----------|
| 4.2.3    | Alternativa 3 . . . . .                         | 53        |
| 4.2.4    | Alternativa 4 . . . . .                         | 54        |
| 4.2.5    | Definição do novo modelo . . . . .              | 55        |
| 4.3      | Adaptações do modelo actual . . . . .           | 58        |
| 4.4      | Resumo ou conclusões . . . . .                  | 59        |
| <b>5</b> | <b>Processo ETL do ADW PRIVATE PRACTICE</b>     | <b>61</b> |
| 5.1      | Estrela dos agendamentos . . . . .              | 61        |
| 5.1.1    | Desenvolvimento . . . . .                       | 61        |
| 5.1.2    | Testes . . . . .                                | 67        |
| 5.2      | Estrela das tarefas dos profissionais . . . . . | 69        |
| 5.2.1    | Desenvolvimento . . . . .                       | 69        |
| 5.2.2    | Testes . . . . .                                | 79        |
| 5.3      | Adaptações do modelo actual . . . . .           | 81        |
| 5.3.1    | Desenvolvimento . . . . .                       | 81        |
| 5.3.2    | Testes . . . . .                                | 83        |
| 5.4      | Resumo ou Conclusões . . . . .                  | 83        |
| <b>6</b> | <b>Conclusões e trabalho futuro</b>             | <b>85</b> |
| 6.1      | Satisfação dos objectivos . . . . .             | 85        |
| 6.2      | Trabalho futuro . . . . .                       | 86        |
|          | <b>Referências</b>                              | <b>90</b> |
| <b>A</b> | <b>Planeamento do projecto</b>                  | <b>91</b> |
| <b>B</b> | <b>Organização da equipa de ADW</b>             | <b>93</b> |

# Lista de Figuras

|      |                                                                                                        |    |
|------|--------------------------------------------------------------------------------------------------------|----|
| 1.1  | Diagrama dos produtos ALERT. . . . .                                                                   | 2  |
| 1.2  | Distribuição do ALERT no mundo. . . . .                                                                | 3  |
| 2.1  | Arquitectura geral de um Data Warehouse. . . . .                                                       | 8  |
| 2.2  | Representação de um modelo em estrela. . . . .                                                         | 10 |
| 2.3  | Dimensão larga. . . . .                                                                                | 11 |
| 2.4  | Representação de um modelo em floco de neve. . . . .                                                   | 13 |
| 2.5  | Processo de desenho dimensional de 4 passos. . . . .                                                   | 15 |
| 2.6  | Edição de um diagrama no Oracle Designer. . . . .                                                      | 22 |
| 2.7  | Abordagem do ODI ao processo ETL. . . . .                                                              | 24 |
| 2.8  | Arquitectura da aplicação ODI. . . . .                                                                 | 25 |
| 3.1  | Tecnologias usadas no desenvolvimento da família de produtos ALERT. . . . .                            | 27 |
| 3.2  | Interface do ALERT PRIVATE PRACTICE. . . . .                                                           | 29 |
| 3.3  | Interface da aplicação ADW. . . . .                                                                    | 31 |
| 3.4  | Visão geral do projecto. . . . .                                                                       | 32 |
| 3.5  | Arquitectura da aplicação ADW. . . . .                                                                 | 34 |
| 3.6  | ADW - Um modelo de dados, vários produtos. . . . .                                                     | 35 |
| 3.7  | Exemplo de criação de um package no ODI. . . . .                                                       | 37 |
| 3.8  | Exemplo de criação de uma interface no ODI. . . . .                                                    | 38 |
| 3.9  | Processo ETL do ADW. . . . .                                                                           | 39 |
| 3.10 | Modelo de detecção de actualizações no ODI. . . . .                                                    | 40 |
| 3.11 | Duração (em segundos) do processo de carregamento da estrela dos profissionais. . . . .                | 45 |
| 4.1  | Estrela dos agendamentos. . . . .                                                                      | 50 |
| 4.2  | Estrela das tarefas dos profissionais - modelo anterior. . . . .                                       | 51 |
| 4.3  | Estrela das tarefas dos profissionais - primeira alternativa. . . . .                                  | 52 |
| 4.4  | Estrela das tarefas dos profissionais - segunda alternativa. . . . .                                   | 53 |
| 4.5  | Estrela das tarefas dos profissionais - terceira alternativa. . . . .                                  | 54 |
| 4.6  | Estrela das tarefas dos profissionais - quarta alternativa. . . . .                                    | 55 |
| 5.1  | Visão geral do processo ETL dos agendamentos. . . . .                                                  | 63 |
| 5.2  | Passos a efectuar pela interface de carregamento da tabela de <i>staging</i> dos agendamentos. . . . . | 65 |
| 5.3  | Passos a efectuar pela interface de carregamento da tabela de factos dos agendamentos. . . . .         | 67 |
| 5.4  | Duração do processo ETL dos agendamentos. . . . .                                                      | 68 |

## LISTA DE FIGURAS

|      |                                                                                                      |    |
|------|------------------------------------------------------------------------------------------------------|----|
| 5.5  | Visão geral do processo ETL das tarefas. . . . .                                                     | 70 |
| 5.6  | Extracção e carregamento de tarefas para a tabela de <i>staging</i> a partir do<br>ALERT®. . . . .   | 72 |
| 5.7  | Carregamento de tarefas para a tabela de factos a partir do ADW. . . . .                             | 74 |
| 5.8  | Passos a efectuar pela interface de carregamento da tabela de <i>staging</i> das<br>tarefas. . . . . | 76 |
| 5.9  | Visão geral do Processo ETL das tarefas agregadas. . . . .                                           | 78 |
| 5.10 | Duração do processo ETL das tarefas dos profissionais. . . . .                                       | 80 |
| 5.11 | Formato do ficheiro CSV da geografia americana. . . . .                                              | 82 |
| A.1  | Diagrama de Gantt do projecto. . . . .                                                               | 91 |
| B.1  | Organização da equipa de ADW. . . . .                                                                | 93 |

# Lista de Tabelas

|     |                                                                     |    |
|-----|---------------------------------------------------------------------|----|
| 2.1 | Comparação entre Sistemas Operacionais e Data Warehouse. . . . .    | 9  |
| 2.2 | Metodologia de tipo 1 para SCD. . . . .                             | 11 |
| 2.3 | Duas metodologias de tipo 2 para SCD. . . . .                       | 12 |
| 2.4 | Metodologia de tipo 3 para SCD. . . . .                             | 12 |
| 3.1 | Informação existente no ADW. . . . .                                | 41 |
| 4.1 | Estrutura das dimensões específicas dos agendamentos. . . . .       | 49 |
| 4.2 | Crescimento das tarefas em relação ao número de episódios. . . . .  | 52 |
| 5.1 | Exemplo de registo da tabela de parametrização das tarefas. . . . . | 71 |

## LISTA DE TABELAS

# Abreviaturas e Símbolos

|        |                                                  |
|--------|--------------------------------------------------|
| ADW    | ALERT Data Warehouse                             |
| BD     | Base de dados                                    |
| BI     | Business Intelligence                            |
| CDC    | Change Data Capture                              |
| CPU    | Central Processing Unit                          |
| CSV    | Comma Separated Value                            |
| CVS    | Concurrent Versions System                       |
| DDL    | Data Definition Language                         |
| DML    | Data Manipulation Language                       |
| ETL    | Extract Transform Load                           |
| FEUP   | Faculdade de Engenharia da Universidade do Porto |
| IDE    | Integrated Development Environment               |
| IKM    | Integration Knowledge Module                     |
| KM     | Knowledge Module                                 |
| LKM    | Loading Knowledge Module                         |
| MIEIC  | Mestrado em Engenharia Informática e Computação  |
| ODI    | Oracle Data Integrator                           |
| OLAP   | Online Analytical Processing                     |
| OLTP   | Online Transaction Processing                    |
| PL/SQL | Procedural Language/Structured Query Language    |
| PP     | PRIVATE PRACTICE                                 |
| SCD    | Slowly Changing Dimension                        |
| SQL    | Structured Query Language                        |
| SVN    | Subversion                                       |
| RDBMS  | Relational Database Management System            |

## ABREVIATURAS E SÍMBOLOS

# Capítulo 1

## Introdução

O presente documento constitui o relatório final do projecto intitulado "Implementação de um mecanismo de extracção e carregamento de dados para o ALERT® DATA WAREHOUSE a partir do ALERT® PRIVATE PRACTICE". O projecto decorreu na ALERT Life Sciences Computing, S.A. de 18 de Fevereiro de 2008 até ao dia 7 Julho de 2008, no contexto do projecto final de curso do David de Almeida Marques, aluno do Mestrado Integrado em Engenharia Informática e Computação da Faculdade de Engenharia da Universidade do Porto. Este Capítulo pretende ser uma introdução geral ao projecto e ao seu contexto, assim como apresentar a estrutura deste relatório. De seguida, são abordados os seguintes temas:

- Contextualização e enquadramento do projecto;
- Apresentação geral do projecto e dos seus principais objectivos;
- Descrição da estrutura do relatório e dos conteúdos de cada Capítulo.

### 1.1 Contexto/Enquadramento

A ALERT Life Sciences Computing, S.A. é a empresa mãe do grupo de empresas ALERT. Está inteiramente dedicada ao desenvolvimento, distribuição e implementação do software clínico ALERT® e está inteiramente dedicada à criação de ambientes clínicos sem papel. Com sede no Porto, a empresa iniciou a sua actividade em Dezembro de 1999. Conta hoje com uma equipa multidisciplinar de cerca de 500 colaboradores permanentes, incluindo clínicos, *designers*, arquitectos, engenheiros, matemáticos e gestores. A missão da ALERT é traduzida pela seguinte frase:

”Melhorar os cuidados de saúde e prolongar a vida, obter lucros no benefício da saúde e inspirar os outros a atingir a excelência da mesma forma.” [AISC08, chap. Introduction]

O ALERT® é uma ferramenta operacional para todos os ambientes de prestação de cuidados de saúde e para todos os profissionais da área da saúde, com a capacidade amplamente demonstrada de produzir ambientes totalmente isentos de papel (ver Figura 1.1). O ALERT® permite a introdução, em tempo real, de toda a informação clínica. É uma aplicação em constante enriquecimento dos seus conteúdos e visa, num futuro próximo, oferecer ferramentas de Inteligência Artificial a todos os profissionais da área da saúde. Na base do ALERT® encontra-se um conceito de fluxo de trabalho<sup>1</sup> que permite o envio contínuo de informação a utilizadores relevantes. O ALERT® interliga as actividades de todos os profissionais de saúde através de conceitos de fluxo de trabalho. Isto permite:

- Apresentar a informação constantemente actualizada, em formato de grelha sumário, para cada perfil de utilizador e utilizador individual;
- Destacar, para cada utilizador, as actividades específicas e áreas de trabalho;
- Alertar cada utilizador para tarefas pendentes.

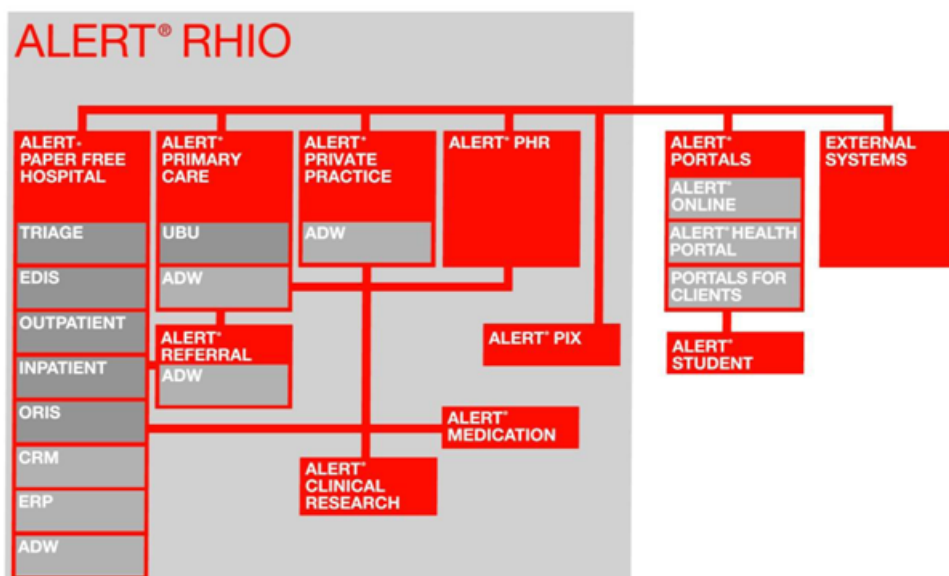


Figura 1.1: Diagrama dos produtos ALERT.

O ALERT® é um produto que está em forte expansão, internacionalmente já foi adoptado em diversos países da Europa e do resto mundo, como por exemplo: Portugal, Espanha, Itália, Holanda, Estados Unidos, Brasil e Malásia (ver Figura 1.2).

<sup>1</sup>workflow



Figura 1.2: Distribuição do ALERT no mundo.

## 1.2 Projecto

A aplicação ALERT® DATA WAREHOUSE (ADW) disponibiliza o acesso, via Web ou a nível local, de estatísticas e dados de apoio à gestão da instituição de saúde. Esses dados são provenientes da aplicação ALERT®.

O projecto incide na extracção de dados da aplicação ALERT® PRIVATE PRACTICE, produto ALERT® para clínicas privadas e consultórios, e no seu carregamento para a aplicação ADW, para posterior tratamento estatístico e visual.

O projecto inclui uma primeira parte de análise sobre os dados mais pertinentes para a gestão financeira, administrativa e clínica num ambiente de medicina privada. Além disso, deverão determinar-se as fontes onde poderão ser recolhidos dados para as estatísticas a apresentar, fase que antecederá a modelação dos dados no *data warehouse*. Com base nesta informação, deverá ser construído um mecanismo de extracção e carregamento de dados. Outro dos objectivos principais do projecto envolve a criação de mecanismos de controlo de qualidade dos dados, de modo a avaliar se as estatísticas geradas vão de encontro ao esperado.

A construção de um sistema de análise de dados para a medicina privada é ainda um mercado inexplorado no nosso país, pelo que o objectivo do projecto contém uma forte componente de inovação. Esta ferramenta visa igualmente o mercado de medicina privada a nível internacional, dado que um dos objectivos da empresa é o seu lançamento via Web.

### 1.3 Motivação e objectivos

Este trabalho aparece devido à inexistência de um produto de *data warehousing* sobre o ALERT® PRIVATE PRACTICE, aplicação clínica que gere o fluxo de dados clínicos numa clínica privada. Um desenvolvimento de um novo produto da família de produtos de Data Warehouse para o ALERT® implica desenvolvimentos de Base de Dados (BD) e da interface com o utilizador. Este projecto incide no desenvolvimento de BD, nomeadamente na criação de um modelo de dados do *data warehouse* e na criação e implementação de um mecanismo de extracção dos dados do sistema ALERT® PRIVATE PRACTICE e carregamento no *data warehouse*.

O trabalho desenvolvido para o ADW PRIVATE PRACTICE tem como principais objectivos:

- Definição de indicadores administrativos, clínicos e financeiros;
- Criação de um novo modelo de dados para o *data warehouse*;
- Criação e implementação de um processo de extracção e carregamento de dados entre o ALERT® PRIVATE PRACTICE e o ADW;
- Garantia da eficiência do processo de extracção e carregamento dos dados;
- Garantia da qualidade de dados.

### 1.4 Estrutura da dissertação

Este documento, para além deste Capítulo introdutório, encontra-se subdividido em vários Capítulos organizados da seguinte forma:

- O Capítulo 2 tem como objectivo apresentar o estado da arte na área de Data Warehouse e ETL (Extract, Transform and Load);
- O Capítulo 3 efectua uma apresentação detalhada do problema e enquadr-o num contexto mais global. São apresentados neste Capítulo os pressupostos, o que se espera obter e quais são os requisitos em que se pode desdobrar;
- A modelação dos dados é descrita no Capítulo 4. Essa descrição envolve uma solução relativa aos problemas analisados no Capítulo 3 que o novo modelo pretende resolver;
- O Capítulo 5 descreve a implementação do processo ETL sobre o modelo de dados especificado no quarto Capítulo. Todo o trabalho realizado na implementação do processo assim como as dificuldades encontradas e respectivas resoluções, são também apresentadas nesse Capítulo;

## Introdução

- As conclusões do trabalho são enumeradas no Capítulo 6. As conclusões do trabalho incluem as principais conclusões, uma avaliação final da solução encontrada e, finalmente, as perspectivas futuras do projecto.

## Introdução

## Capítulo 2

# Estado da Arte

Neste Capítulo são apresentados os conceitos de Data Warehouse, ETL, e das diferentes áreas circundantes. Alguns dos conceitos enunciados a seguir fazem parte da área de Base de Dados, nomeadamente Base de Dados com modelos relacionais. Este Capítulo parte do pressuposto que os conceitos principais de Sistema de Gestão de Base de Dados Relacionais (RDBMS) são conhecidos. Para mais informações sobre conceitos introdutórios de base de dados, ver [Dat99], [UW97] ou [GMWU99].

### 2.1 Data Warehouse: conceitos gerais

De acordo com Ralph Kimball, um *data warehouse* é:

*”A data warehouse is a system that extracts, cleans, conforms, and delivers source data into a dimensional data store and then supports and implements querying and analysis for the purpose of decision making.”* [KC04, chap. Surrounding the Requirements]

O termo *data warehouse* além de se referir ao sistema de uma forma global é vulgarmente associado ao nome da Base de Dados que vai conter os dados.

Os principais objectivos na construção de um *data warehouse* são:

- Ser um local de armazenamento de uma grande quantidade de dados;
- Representar informação presente noutras fontes de dados;
- Permitir um acesso fácil à informação de uma organização;
- Ser consistente com os sistemas fonte de forma a apresentar dados coerentes;
- Servir de base para as decisões futuras da organização sobre a qual incide o *data warehouse*;

- Ser adaptável e preparado para qualquer mudança que possa ser efectuada.

A arquitectura de um Data Warehouse pode ser vista na Figura 2.1 da página 8. Esta arquitectura contém 4 componentes:

- Sistema operacional;
- Processo ETL;
- Data Warehouse;
- Software de visualização.

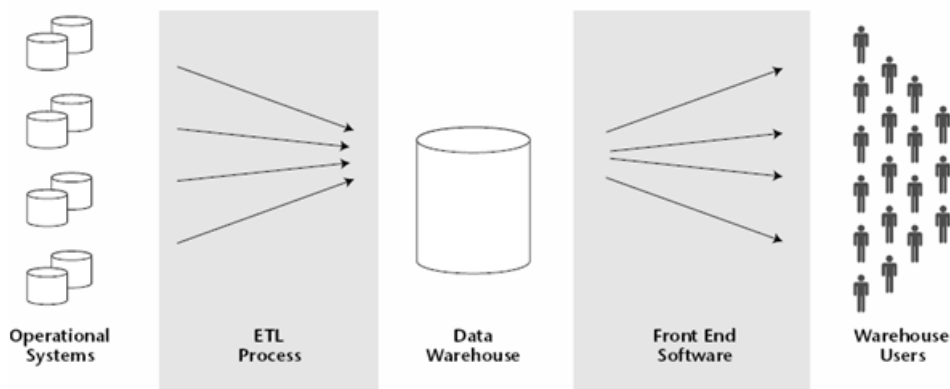


Figura 2.1: Arquitectura geral de um Data Warehouse.

Um **Sistema Operacional** é uma aplicação que suporta a execução de um processo de negócio, que regista a actividade do processo de negócio e cuja principal função é servir de sistema de registo. Neste projecto o sistema operacional é a base de dados da aplicação ALERT® PRIVATE PRACTICE. Dado que o sistema operacional é um sistema transaccional, essa base de dados encontra-se, tipicamente, normalizada<sup>1</sup> num modelo relacional. A arquitectura permite efectuar operações de actualização e inserção de forma rápida pois os dados são guardados de forma atómica.

Um **Data Warehouse**, no sentido mais literal da palavra, é uma Base de Dados que suporta a avaliação, ou medição, dos processos de negócio. Geralmente implementada usando um modelo dimensional, a BD consiste numa cópia organizada de forma característica, de modo a suportar processos analíticos.

O processo **ETL (Extract, Transform, Load)** é o mecanismo que permite mover dados para um Data Warehouse. Esse mecanismo pode ser automático ou manual e pode usar ferramentas avançadas ou consistir em sequências de código SQL. O processo contém três passos importantes cuja ordem pode variar e cuja separação pode ser mais ou menos evidente, tanto a nível físico como do ponto de vista da modelação. A

<sup>1</sup>Geralmente na terceira forma normal - ver [RG03, chap. Schema refinement and normal forms]

fase de extracção corresponde à extracção dos dados das tabelas da origem. A fase de transformação corresponde à reorganização dos dados num modelo apropriado, um dos modelos mais usados e adequados de armazenamento de dados num *data warehouse*. A fase de carregamento corresponde à fase de inserção dos dados no *data warehouse*.

O *software* de visualização, muitas vezes sob a forma de uma aplicação de **Reporting**, é a aplicação que consome os dados do *data warehouse* e que apresenta os resultados em diversos formatos. O objectivo principal é servir de interface com a BD, transformando as pesquisas do utilizador em linguagem SQL e transformando os resultados da BD num formato de visualização apropriado como relatórios, gráficos, entre outras formas.

Algumas das principais diferenças entre sistemas de *data warehouse* e sistemas operacionais são apresentadas na tabela 2.1.

Tabela 2.1: Comparação entre Sistemas Operacionais e Data Warehouse.

|                         | Sistemas Operacionais                 | Data Warehouse                      |
|-------------------------|---------------------------------------|-------------------------------------|
| Também conhecido como   | On Line Transaction Processing (OLTP) | Online Analytical Processing (OLAP) |
| Objectivo               | Execução de um processo de negócio    | Avaliação de um processo de negócio |
| Tipo de interacção      | Insert, Update, Query, Delete         | Query                               |
| Âmbito de interacção    | Transacções individuais               | Transacções agregadas               |
| Padrões de interrogação | Previsível e estável                  | Imprevisível e instável             |
| Foco temporal           | Corrente                              | Corrente e histórico                |
| Tipo de modelo de dados | Normalizado                           | Desnormalizado                      |

## 2.2 Modelo de dados de um Data Warehouse

Um dos principais modelos de representação dos dados num *data warehouse* é o **modelo em estrela**. O nome deste modelo advém do facto da sua representação gráfica ser semelhante à representação de uma estrela. Ralph Kimball popularizou essa aproximação ao modelo de dados de um *data warehouse* nos anos 90. A partir do seu trabalho e dos seus livros, Kimball estabeleceu uma terminologia padrão que é agora usada em todo o mundo para desenhar e construir *data warehouses*. Com o co-autor Margy Ross, Kimball disponibiliza um tratamento detalhado desses princípios, na obra *The Data Warehouse Toolkit* [KR02]. Um exemplo de um modelo de dados organizado num modelo em estrela é apresentado na Figura 2.2.

Num modelo em estrela existem basicamente duas entidades que armazenam dois tipos distintos de informação:

- Factos.
- Dimensões.

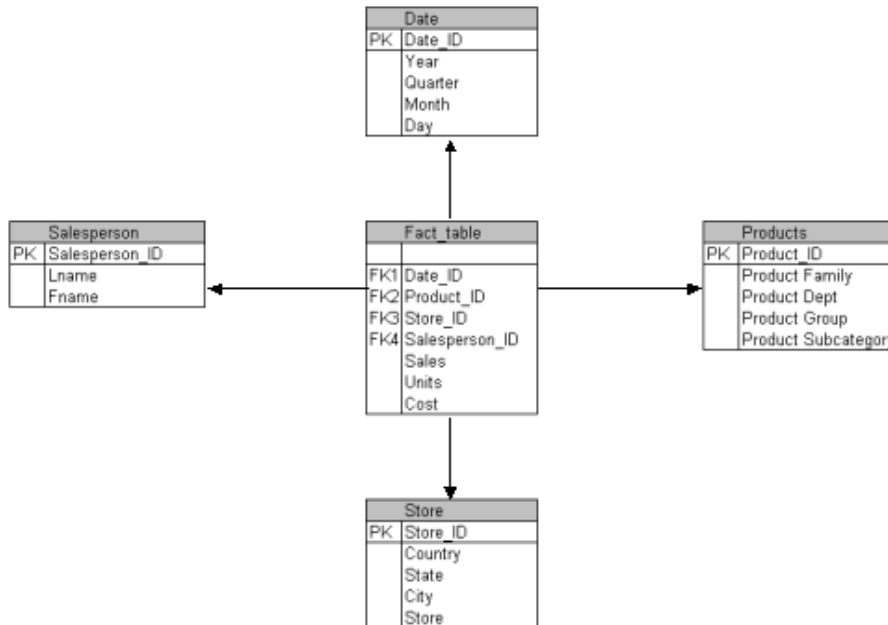


Figura 2.2: Representação de um modelo em estrela.

As tabelas de factos representam todas as medições, contagens, cálculos sobre a informação do sistema operacional. Uma tabela de factos representa a avaliação sobre um processo de negócio e tipicamente tem como dados valores numéricos como custos e quantidades.

Tipicamente, uma tabela de **factos** contém dados numéricos. De forma a contextualizar esses dados existem as dimensões que permitem dar a esses dados o seu contexto. Uma **dimensão** contém informação textual que permite descrever a entidade que representa. Quanto mais informação tiver uma dimensão, mais rico é o *data warehouse*, pois existe imensa informação que permite contextualizar as medições recolhidas. De forma a relacionar factos com dimensões, é usual guardar chaves estrangeiras para as dimensões dentro das tabelas de factos. De uma forma geral, as dimensões são tabelas com muitas colunas e os factos são tabelas com muitas linhas relativamente às colunas. Esse aspecto advém do facto de no caso das dimensões querermos a maior quantidade possível de informação para cada registo. No caso da tabela de factos o seu crescimento em termos de registos costuma ser muito maior do que o número de colunas que pode ter. As dimensões são consideradas *largas* enquanto que os factos são considerados *profundos*. Um exemplo de uma dimensão larga é apresentado na Figura 2.3 [KRT<sup>+</sup>98].

Existem vários tipos de dimensões. Dimensões estáticas não mudam ao longo do tempo e **Slowly Changing Dimensions** (SCD) incluem informação que muda *lentamente*. Para lidar com a mudança de informação numa dimensão, existem várias possibilidades designadas habitualmente por *tipos SCD*. Os tipos mais frequentemente usados são o 1, 2

| DAY                |
|--------------------|
| day_key            |
| full_date          |
| day_of_week_number |
| day_of_week_name   |
| day_of_week_abbr   |
| day_of_month       |
| holiday_flag       |
| weekday_flag       |
| weekend_flag       |
| month_number       |
| month_name         |
| month_abbr         |
| quarter            |
| quarter_month      |
| year               |
| year_month         |
| year_quarter       |
| fiscal_period      |
| fiscal_year        |
| fiscal_year_period |

Figura 2.3: Dimensão larga.

e o 3 mas existem ainda os tipos 0, 4, 5 e o 6.

A **metodologia de tipo 1** escreve por cima da informação antiga, e portanto não guarda informação histórica. Esta metodologia é a mais apropriada quando se corrige algum tipo de erros de dados, como por exemplo a ortografia de um nome (assumindo que nunca será preciso saber como estava escrito antes).

Outro exemplo pode ser uma BD que guarda informação dos fornecedores. Se o fornecedor mudar as suas instalações de país, a tabela actualizada iria escrever simplesmente por cima do antigo registo. Esse exemplo está representado na tabela 2.2.

Tabela 2.2: Metodologia de tipo 1 para SCD.

| Chave | Nome               | País     |
|-------|--------------------|----------|
| 001   | Fornecedor Exemplo | Portugal |
| 002   | ...                | ...      |

| Chave | Nome               | País   |
|-------|--------------------|--------|
| 001   | Fornecedor Exemplo | Taiwan |
| 002   | ...                | ...    |

A desvantagem clara deste método é a de não guardar registos históricos no *data warehouse*. Não podemos saber por exemplo que um fornecedor mudou de país mas em contrapartida existe a vantagem de ser um método muito fácil de implementar e manter.

A **metodologia de tipo 2** permite guardar registos históricos criando múltiplos registos nas dimensões com chaves diferentes. Com a metodologia de tipo 2, temos preservação do histórico de forma ilimitada já que um novo registo é inserido a cada mudança. Voltando ao exemplo descrito acima, podemos usar um de dois métodos exemplificados na tabela 2.3. O primeiro método simplesmente acrescenta um campo *Versão* à tabela de forma a ter todas as versões de um registo de forma ordenada. Outro método possível usa

datas de início e de fim de forma a saber o intervalo de tempo em que o registo esteve activo. O registo actual corresponde àquele que tiver uma data de fim muito grande ou com valor *null* segundo o caso que se preferir implementar.

Tabela 2.3: Duas metodologias de tipo 2 para SCD.

| Chave | Nome               | País     | Versão |  |
|-------|--------------------|----------|--------|--|
| 001   | Fornecedor Exemplo | Portugal | 0      |  |
| 002   | Fornecedor Exemplo | Taiwan   | 1      |  |

| Chave | Nome               | País     | Data de início | Data de fim |
|-------|--------------------|----------|----------------|-------------|
| 001   | Fornecedor Exemplo | Portugal | 01-Jan-2000    | 21-Dec-2004 |
| 002   | Fornecedor Exemplo | Taiwan   | 22-Dec-2004    | <i>null</i> |

No caso de actualizações frequentes, a metodologia de tipo 2 tem a desvantagem de permitir múltiplas chaves anónimas. Efectivamente, quando um registo é actualizado, é necessário actualizar todos os registos que apontam para esse registo e que pretendem ter a informação mais recente. Este tipo de actualizações pode ser complexo e operações de alto custo numa BD pelo que actualizações de tipo 2 não são recomendadas em caso de dados com actualizações muito frequentes.

A **metodologia de tipo 3** pretende guardar o histórico mas de forma limitada alterando a estrutura da dimensão. O tipo 3 adiciona colunas à dimensão de forma a guardar o valor que mudou e ao mesmo tempo o novo valor. Esta metodologia é muito limitada já que se ocorrerem duas mudanças seguidas por exemplo, só é guardada a última alteração. Esta metodologia é exemplificada usando novamente o exemplo da tabela dos fornecedores na tabela 2.4.

Tabela 2.4: Metodologia de tipo 3 para SCD.

| Chave | Nome               | País original | Data de efectivação | País actual |
|-------|--------------------|---------------|---------------------|-------------|
| 001   | Fornecedor Exemplo | Portugal      | 22-Dec-2004         | Taiwan      |

Um caso prático, tipicamente escolhido para explicar o conceito de factos e dimensões, é o caso do retalho. Este caso está representado pela Figura 2.2 que representa um modelo em estrela. Temos um processo de negócio típico que representa as vendas e que pode ser representado por uma tabela de factos. Essa tabela de factos armazena, por exemplo, cada registo de uma transacção efectuada de um determinado produto. A cada registo está associado o número de unidades vendidas, o custo e lucro da transacção. De forma a contextualizar a transacção, existem quatro chaves estrangeiras correspondentes às quatro dimensões: Produto, Data, Vendedor e Loja. Desta forma, para cada transacção podemos saber o produto, a data -num formato com grande detalhe- em que foi vendido, a loja, o vendedor e informações sobre o vendedor.

Este tipo de **cruzamento entre factos e dimensões** é o método natural de obter informações num *data warehouse* e constitui a base de trabalho sobre um *data warehouse*. A dimensão da loja, por exemplo, tem como atributos o país, estado, cidade e nome da loja. Esta informação permite enriquecer o conhecimento sobre uma transacção que foi efectuada e permite o cruzamento de informação de uma forma ilimitada. Com este modelo de dados, é possível pesquisar por todas as transacções num determinado país ou numa determinada cidade.

Este tipo de **agregação** de informação é também característico de um *data warehouse* e permite fazer um tratamento estatístico a larga escala até ao nível mais baixo. A diversidade de perguntas é grande e um modelo em estrela bem desenhado e construído permite-nos responder virtualmente a qualquer pergunta.

No exemplo acima foi necessário definir o nível de detalhe de cada registo da tabela de factos. Esse nível de detalhe poderia ter sido definido doutra forma, como por exemplo definindo um registo como sendo uma transacção efectuada numa loja. Mas aí o problema de existirem vários produtos diferentes envolvidos em cada transacção não poderia ser resolvido. O nível de detalhe de cada linha de uma tabela de factos chama-se **granularidade**. Até aqui destacaram-se os benefícios do modelo em estrela para o desenho de um modelo de dados num *data warehouse* mas existem outros modelos que podem ser adoptados consoante as necessidades. Um dos modelos alternativos é o modelo em floco de neve. Mais uma vez o nome advém da sua representação gráfica ser semelhante a um floco de neve. Na Figura 2.4, podemos ver um exemplo de um modelo em floco de neve.

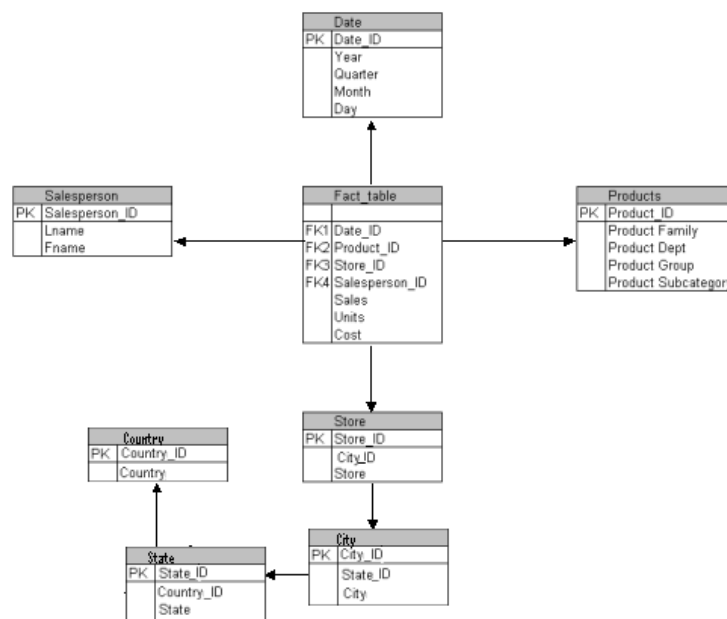


Figura 2.4: Representação de um modelo em floco de neve.

O **modelo em floco de neve** é essencialmente uma extensão do modelo em estrela em que existe uma maior normalização das dimensões do modelo. Esta normalização pretende eliminar a redundância que existe no modelo em estrela. De facto, o modelo em estrela, é bastante desnormalizado. Isto significa que, por exemplo, voltando ao caso prático descrito acima, a dimensão da loja que contém um atributo "país" tem o mesmo nome de país repetido nos registos da dimensão, no caso de existirem muitas lojas localizadas no mesmo país. E da mesma forma, a mesma cidade e o mesmo estado aparecem de forma repetida em registos diferentes. Esse problema é resolvido no modelo relacional com a terceira forma normal em que um modelo é totalmente normalizado de forma à evitar a replicação de informação o que permite maximizar a eficiência no momento de operações DML (*insert*, *update*, *delete*) e evitar inconsistências na base de dados. Algumas das vantagens que um modelo em floco de neve proporciona são:

- Melhoramento na consistência e manutenção da informação já que não há redundância;
- Melhoramento no espaço ocupado já que não há replicação de informação.

Contudo, esses melhoramentos necessitam de ser realmente avaliadas no sistema que está a ser desenvolvido. Efectivamente, na maior parte dos casos as tabelas que representam as dimensões têm um tamanho muito inferior do que as tabelas de factos pelo que o ganho de espaço pela normalização das tabelas muitas vezes não ultrapassa 1% do espaço total. Da mesma forma, na maior parte dos sistemas é preferível ter um só cruzamento entre facto e dimensões do que vários cruzamentos com muita profundidade.

## 2.3 Construção de um Data Warehouse

Na secção anterior descreveu-se a modelação de um processo de negócio pelo modelo em estrela com uma tabela de factos ligada a várias dimensões. Tipicamente uma organização possui mais do que um processo de negócio e por consequente precisa de avaliar mais do que um processo de negócio no mesmo *data warehouse*. Cada processo de negócio modelado num *data warehouse* é tipicamente denominado de **data mart**, ou **estrela**. [Kim97]

Um *data warehouse* é uma soma de data marts permitindo, assim, representar todos os processos de negócio de uma organização numa só base de dados. Desta forma, um *data warehouse* pode ser construído de forma incremental, tendo a possibilidade de adicionar um novo processo de negócio a qualquer momento. Cada data mart contém informação extremamente útil sobre o seu processo de negócio mas de forma isolada.

Um *data warehouse* é muito mais poderoso se se conseguir ligar a informação de diferentes data marts, ou seja, diferentes processos de negócio. Um *data warehouse* que consegue relacionar os seus data marts é essencial de forma a poder responder a perguntas

que incidem sobre vários processos de negócio. Para conseguir relacionar os data marts, uma prática popular é a utilização de **dimensões conformes**. [Inm96] Uma dimensão conforme:

- significa o mesmo em todas as tabelas de factos;
- chave definida e anónima;
- dados tratados e consistentes.

Um conjunto de dimensões conformes, ou *warehouse bus*, deve ser definido o mais cedo possível, na construção de um *data warehouse*, de forma a permitir desde o início informação consistente e relacionável em todo o *data warehouse*.

Com a ajuda de dimensões conformes é possível construir um *data warehouse* a partir da construção de data marts. A construção de uma estrela segundo o método de 4 passos de Kimball [KR02, chap. 2] é apresentada na Figura 2.5



Figura 2.5: Processo de desenho dimensional de 4 passos.

O primeiro passo consiste na **escolha do processo de negócio** que se pretende modelar para o *data warehouse*. Essa escolha deve ser ponderada com os requisitos de negócio assim como com a análise da informação disponível no sistema operacional.

A **definição da granularidade** é o segundo passo na construção do data mart. Esse passo é de extrema importância pois é esse que vai definir o nível de detalhe da estrela que estamos a desenhar. Deve-se escolher preferencialmente a informação mais atómica possível do processo de negócio. Informação atómica é a informação mais detalhada, isto é, a informação não pode ser dividida a um nível mais baixo. Desta forma, garantimos toda a informação possível.

No passo seguinte é necessário **escolher as dimensões**. Tipicamente, uma definição da granularidade efectuada com cuidado permite definir de forma quase directa as dimensões a considerar. Além das dimensões que provêm da definição da granularidade é possível adicionar dimensões que tipicamente correspondem a um só valor da combinação das dimensões primárias.

Finalmente, o quarto passo corresponde à **definição dos factos**. A definição dos factos corresponde à escolha dos atributos (ou medidas) que irão fazer parte da tabela de

factos. Tipicamente, essa escolha está condicionada aos dados que estão efectivamente disponíveis no sistema operacional. Existem diferentes tipos de medidas para uma tabela de factos:

- Medidas aditivas são medidas que podem ser somadas sobre todas as dimensões;
- Medidas semi-aditivas são medidas que podem ser somadas sobre algumas dimensões e não sobre outras;
- Medidas não-aditivas são medidas que não podem ser somadas.

Como vimos nesta secção, os princípios do desenho dimensional apontam para que a granularidade da tabela de factos seja a que tiver o maior nível de detalhe possível. Esse princípio garante que será possível apresentar os factos em qualquer contexto dimensional desejado. No entanto, na maior parte dos casos, as queries dos utilizadores do sistema não interrogam essas medidas atómicas. Pelo contrário, certos grupos de medidas serão agregados. Se é verdade que as medidas atómicas raramente aparecem nos resultados finais, devem estar disponíveis no RDBMS para calcular as agregações. [Ada06]

As **tabelas de agregação** procuram melhorar a performance das queries, reduzindo a quantidade de dados que têm que ser acedidos. Pelo meio de pré-agregação dos dados numa tabela de factos, é possível reduzir a quantidade de trabalho que um RDBMS tem de realizar para responder a uma query. Basicamente, estamos a aumentar a performance, reduzindo o número de linhas que têm de ser acedidas.

As tabelas de agregação podem, se bem utilizadas, ser um instrumento poderoso para melhorar a performance do *data warehouse*. No entanto, é necessário ter em atenção alguns perigos na utilização de tabelas de agregação. As tabelas de agregação representam sumários de informação, pelo que nem todas as agregações possíveis devem ser feitas. É necessário avaliar as agregações importantes à guardar de forma persistente e pré-calculadas. Podem existir, por exemplo, agregações com um elevado custo de processamento e que raramente querem ser vistas pelos utilizadores, pelo que a quantidade de trabalho de fazer sempre o cálculo e guardá-lo na base de dados pode não compensar. Outro cuidado especial, a ter em conta quando se constrói tabelas de agregação, é verificar a consistência da informação. Efectivamente, o sumário de informação representada pela agregada tem de representar **exactamente** a mesma informação da tabela de factos da qual a agregada provém. Essa transparência nos resultados permite construir mecanismos de invisibilidade para o utilizador, de forma a que uma query normal chame a agregada em vez de chamar a tabela de factos.

Como caso prático, voltando ao exemplo que vimos na secção 2.2, tínhamos uma tabela de factos que correspondia a cada transacção feita numa loja. Uma agregada simples, nesse caso, seria essa mesma tabela de factos agregada ao mês. Desta forma, podemos

saber, acedendo directamente a essa tabela, quantas unidades são vendidas num determinado mês. Contudo, se quiséssemos ver o detalhe das vendas de um dia desse mês tínhamos que aceder directamente à tabela de factos, pois essa informação já não existe na agregada. Este tipo de decisões sobre a tabela a aceder para responder à query deve ser transparente ao utilizador.

O exemplo acima é relativo a uma **agregada invisível**, que é o mais corrente mas existem vários tipos de agregadas. Os outros tipos de agregadas que existem são:

- **Agregada pré-cruzada:** Uma agregada pré-cruzada combina um facto agregado e os atributos de uma dimensão numa só tabela. Se é verdade que pode aumentar a performance de resposta às queries, uma agregada pré-cruzada tem a tendência de ocupar espaço excessivo em disco.
- **Tabela derivada:** As tabelas derivadas são um grupo de sumários que procura melhorar a performance alterando a estrutura da tabela de factos sumária ou alterando o âmbito do seu conteúdo. Um exemplo de tabela derivada é a tabela de factos fundida (*merged*). Essa tabela de factos combina factos de diversas tabelas ao mesmo nível de granularidade.
- **Tabela com novos factos:** O último tipo de tabela sumária é paradoxal já que não contém atributos presentes nas tabelas originais. Este tipo de sumários ocorre quando um facto não exhibe características de aditividade. Novos factos são criados como agregação sobre algumas das dimensões e sobre factos da tabela de factos de origem, mudando assim a granularidade.

## 2.4 Processo ETL

Um processo **ETL (Extract-Transform-Load)** é fundamental num *data warehouse*. Um sistema ETL bem desenhado extrai a informação dos sistemas fontes, reforça a qualidade e a consistência da informação, normaliza os dados de forma a que diversos sistemas fontes possam ser utilizados, e finalmente carrega os dados num formato pronto para apresentação de tal forma que os programadores da aplicação possam construir aplicações e utilizadores finais possam tomar decisões. [KC04]

O sistema ETL adiciona um valor significativo aos dados do *data warehouse* e não é só um meio de transferência de informação. De forma mais específica o sistema ETL tem como objectivos:

- Remover os erros e corrigir dados em falta;
- Disponibilizar medidas de confiança nos dados;
- capturar o fluxo de dados transaccional para salvaguarda;

- Ajustar informação de múltiplas fontes de forma a poder ser usada em conjunto;
- Disponibilizar estruturas de dados usáveis por ferramentas de exploração de dados.

Cada passo do processo ETL tem diferentes objectivos. A fase de **extracção** inclui de forma geral:

- Ler os modelos de dados de origem;
- Ligar-se e aceder aos dados;
- Agendar a extracção de dados do sistema fonte;
- Detectar a actualização de dados;
- Colocar os dados extraídos numa área de *staging*.

Um dos aspectos mais complexos das tarefas descritas acima é a **detecção da alteração dos dados**. Tipicamente num *data warehouse* típico, irá ocorrer um carregamento inicial de todos os dados do sistema fonte a um determinado ponto no tempo. Depois desse carregamento ocorrer não queremos reprocessar dados que já foram carregados e não foram alterados. Desta forma, o processamento total dos dados do sistema fonte é uma tarefa com demasiado custo para ser considerada na maior parte dos sistemas. Acabávamos por sobrecarregar os sistemas fonte e os canais de comunicação devido ao excesso de informação. Uma solução é a detecção de actualização de dados. Este mecanismo permite-nos detectar quais os registos que foram alterados desde o último carregamento de forma a processar exclusivamente esses registos. Algumas das possibilidades de implementação de um mecanismo de detecção de alteração dos dados são:

- Triggers nos sistemas fontes;
- Colunas de auditoria;
- Log miners;
- Processo de eliminação (comparar tabela actual com última tabela carregada).

A fase de **transformação** envolve, principalmente, tarefas de limpeza e de normalização dos dados. as principais tarefas correspondentes são:

- Restrições de propriedades de colunas;
- Restrições da estrutura;
- Restrições de regras de dados e valores;
- Restrições de regras complexas de negócio;

- Normalizar os conteúdos das dimensões;
- Normalizar as métricas e indicadores (das tabelas de factos);
- Eliminar a redundância;
- Internacionalizar os dados;
- Colocar os dados transformados numa área de *staging*.

Por último, a fase de **carregamento** envolve, na maior parte dos casos, as seguintes tarefas:

- Comparar os dados actualizados com os dados existentes nas *slowly changing dimensions*;
- Carregar os tipos 1, 2 e 3 nas *slowly changing dimensions*;
- Comparar os dados actualizados com os dados existentes nos factos;
- Inserir dados novos nos factos;
- Actualizar dados já existentes nos factos;
- Cruzar as dimensões e tabelas de *staging* de factos de forma a carregar as chaves primárias nas chaves estrangeiras das tabelas de factos;
- Carregar e actualizar as tabelas de agregação;
- Colocar os dados carregados numa área de *staging*.

Em todas as fases descritas, existe a hipótese de colocar os dados carregados numa área de *staging*. Uma **área de *staging*** armazena dados que estão *à caminho* da área de apresentação final do *data warehouse*. A decisão de colocar ou não os dados numa área de *staging*, a cada etapa depende, do ambiente e dos requisitos de negócio. Na maior parte dos casos, existe pelo menos uma área de *staging* no processo ETL de um *data warehouse*.

Uma das mais-valias possíveis da utilização de áreas de *staging* é a **recuperação de dados**. Na maior parte dos ambientes empresariais, é uma boa prática guardar os dados numa *staging* logo à seguir a sua extracção do sistema fonte e a seguir a cada transformação significativa dos dados. Essas áreas de *staging*, quer numa base de dados ou num sistema de ficheiros, servem de pontos de recuperação. Implementadas essas tabelas, o processo não terá que interrogar o sistema fonte novamente se uma transformação falhar ou transformar novamente os dados.

O **Backup** dos dados também pode ser optimizado através da utilização de áreas de *staging*. Na maior parte dos casos, o grande volume de informação impede um *data warehouse* de ter cópias regulares de segurança ao nível da base de dados. As tabelas de

*staging* podem ser guardadas, ao nível do sistema de ficheiros, comprimidas e arquivadas de forma a minimizar o espaço ocupado. Desta forma podemos guardar grandes quantidades de dados e à qualquer momento podemos recarregar o *data warehouse*, a partir de um certo ponto no tempo.

Finalmente, a utilização de áreas de *staging* são fundamentais, ao nível de **Auditoria**. É muitas vezes difícil relacionar dados entre a fonte e destino já que se perde essa relação no código ETL. De forma a fazer auditorias, ou até debug, do processo ETL, é interessante ter áreas de *staging* em várias fases do processo ETL. Desta forma é muito mais fácil e directo a comparação entre, por exemplo, um ficheiro de entrada e transformações e ficheiros de saída. Essas áreas de *staging* são especialmente úteis quando o sistema fonte escreve por cima do seu próprio histórico. Desta forma, quando questões sobre a integridade da informação do *data warehouse* surgem dias ou semanas depois de um evento ter ocorrido, a utilização de áreas de *staging* desse período de tempo pode restaurar a confiança no *data warehouse*.

## 2.5 Tecnologias

Na área de *data warehouse*, existem diversas soluções para responder às diversas necessidades, tanto ao nível do motor de base de dados como da interface para o utilizador ou ferramentas de integração. Este projecto está limitado no que diz respeito à escolha de ferramentas a utilizar já que pretende integrar-se numa aplicação já existente, o ADW. As ferramentas que são apresentadas a seguir foram usadas no decorrer do projecto ou tiveram um impacto no seu desenvolvimento.

### 2.5.1 Oracle



O **Oracle 10g Enterprise Edition** é um sistema de gestão de bases de dados relacionais conceituado. É usado a nível internacional com provas dadas quer pela sua estabilidade e performance, quer pela sua polivalência. Além disso, é um sistema que dá grandes garantias de compatibilidade com um elevado número de ferramentas.

Uma das características importantes das base de dados Oracle é a possibilidade de correr procedimentos e funções dentro da própria base de dados. Esses procedimentos e funções podem ser escritos em PL/SQL (A linguagem procedural proprietária da Oracle e que é uma extensão do SQL), ou a linguagem orientada para objectos Java. Essas linguagens são muito mais poderosas do que o SQL simples e permitem executar processos muito mais complexos e manter um alto nível de performance, devido ao facto da execução ser optimizada pelo próprio motor da Oracle.

Outro dos aspectos fundamentais da sua arquitectura é separar o armazenamento lógico e o armazenamento físico dos dados. Assim as *tablespaces* armazenam os dados de forma lógica enquanto que os dados físicos são guardados em *data files*. Existem outros tipos de ficheiros físicos como os *control files*, para armazenar as configurações da base de dados e os *redo log files*, que contêm informação para recuperação de dados entre outros.

Outras funcionalidades importantes das base de dados Oracle mas que não irão ser descritas são:

- Query optimizer;
- Data dictionary;
- Materialized view;
- RAC (Real Application Clusters);
- Partitioning.

### 2.5.2 PL/SQL Developer



O **PL/SQL Developer** é um IDE (Integrated Development Environment) que está especificamente centrado no desenvolvimento sobre bases de dados Oracle. Este ambiente de desenvolvimento integrado tem uma importante quota de mercado à nível internacional.

O PL/SQL Developer concentra-se sobretudo na usabilidade, qualidade do código e produtividade, já que são aspectos chave durante o desenvolvimento de aplicações sobre plataformas da Oracle. Algumas das principais funcionalidades do PL/SQL Developer são enumeradas a seguir:

- Editor poderoso de PL/SQL e SQL;
- Debugger integrado;
- Geração de relatórios e diagramas;
- Ferramentas de gestão de projectos;
- Criação de queries de forma gráfica;
- Ferramentas de exportação e geração de scripts.

### 2.5.3 Oracle Designer

O **Oracle Designer** é uma ferramenta dedicada à criação de diagramas sobre modelos de dados assim como à própria modelação de dados. A ferramenta é desenvolvida pela

Oracle Corporation mas é compatível com diversas plataformas. A ferramenta permite gerar e editar diagramas a partir de modelos de dados existentes e gerar modelos de dados a partir da criação de diagramas. É uma ferramenta que usada de forma correcta pode ser uma ajuda muito poderosa no desenho de modelo de dados. Um exemplo de edição de um diagrama no Oracle Designer é apresentado na Figura 2.6 da página 22.

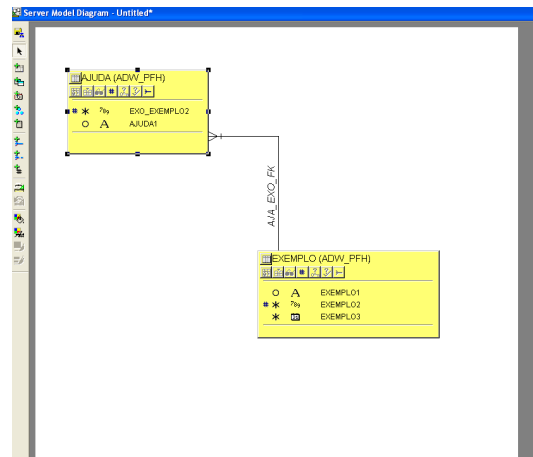


Figura 2.6: Edição de um diagrama no Oracle Designer.

O Oracle Designer é uma ferramenta bastante flexível na edição dos diagramas permitindo a personalização total dos elementos apresentados nos diagramas. Alguns dos elementos ou objectos disponíveis na geração de diagramas são:

- Tabelas;
- Colunas;
- Comentários das colunas;
- Indexes;
- Chaves (Primary Key ou Foreign Key);
- Constraints.

#### 2.5.4 MicroStrategy

**MicroStrategy** O MicroStrategy é uma plataforma que fornece soluções de Business Intelligence de elevada qualidade. É reconhecido como um dos líderes de mercado, e tem como clientes alguns dos mais prestigiados grupos norte-americanos em diversas áreas de indústria e serviços. O *Magic Quadrant* para aplicações de Business Intelligence em 2008 publicado pela Gartner considera a MicroStrategy uma das plataformas com maior capacidade de execução e maior visão global. [Sch08] As principais funcionalidades do MicroStrategy são :

- **Dashboards:** Relatórios em formato gráfico que permite uma visualização imediata e atractiva da informação importante por partes de gestores.
- **Enterprise Reporting:** Relatórios num formato detalhado que permite a visualização de toda a informação disponível por parte de todos os utilizadores do negócio.
- **OLAP Analysis:** Análises avançadas da informação com capacidade de filtrar, cruzar, ordenar, reorganizar a informação de forma a permitir aos gestores fazer análises que ultrapassam a simples visualização de relatórios.
- **Advanced & Predictive Analysis:** permite fazer interrogações ao *data warehouse* até ao nível de uma transacção de forma a dar a utilizadores avançados a possibilidade fazer análise estatística e preditiva.
- **Alerts & Proactive Notification:** Entrega de informação a um grande número de utilizadores pelo meio de agendamentos, excepções de negócio ou pedidos.

O MicroStrategy permite ainda ter diversos *data warehouses* como fontes juntando-os numa metadata única e permitindo numa só interface Web a visualização de vários sistemas origem. A Oracle e a MicroStrategy são parceiros globais o que possibilita uma certificação total da MicroStrategy para uma vasta gama de produtos da Oracle e uma garantia de uma integração total entre as duas plataformas.

### 2.5.5 Oracle Data Integrator

O **Oracle Data Integrator(ODI)** é uma ferramenta multi-plataforma de integração de sistemas de informação e que disponibiliza uma automatização do processo ETL. Anteriormente denominado de Sunopsis, entretanto comprado pela Oracle, o ODI é uma plataforma escrita em Java e que permite um alto grau de flexibilidade e modularidade. Uma das principais vantagens competitivas do ODI são:

- **Performance:** Grande performance devido à uma abordagem única ao processo ETL com uma arquitectura ELT;
- **Produtividade:** Elevado nível de produtividade graças aos vários módulos gráficos que permitem desenvolver de forma mais rápida e com uma manutenção mais eficiente;
- **Integração:** Processo de detecção de mudanças automatizado compatível com qualquer RDBMS (Sistema de gestão de base de dados relacional);
- **Modularidade:** Existência de *Knowledge Modules* que permitem o suporte a qualquer base de dados ou aplicação.

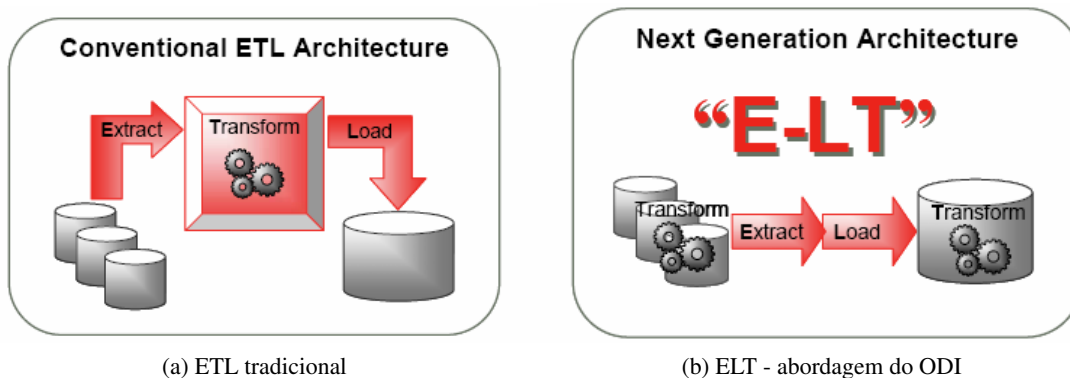


Figura 2.7: Abordagem do ODI ao processo ETL.

Uma das principais características do ODI, relativamente aos seus concorrentes, é a sua abordagem única ao processo de ETL. A sigla ETL representa as diversas fases de um processo de extracção, transformação e carregamento mas também a ordem tradicional pela qual são executados esses processos. O ODI introduz uma abordagem nova em que o carregamento é feito antes da transformação dos dados de forma a maximizar a eficiência dos recursos disponíveis. [Kar]

Essa abordagem **ELT** permite aproveitar o poder da base de dados de destino para tratar das operações de transformação que geralmente são as operações com mais custo a nível de CPU. Permite também minimizar o número de carregamentos: temos um carregamento só em vez de dois no modelo tradicional. Aumentamos desta forma a performance e reduzimos os custos já que não precisamos de um servidor dedicado para o ETL. A Figura 2.7b representa a abordagem seguida pelo ODI no que diz respeito ao processo ETL sob a forma de ELT.

Os **Knowledge Modules** do ODI, ou KMs, são os núcleos da arquitectura de integração do produto. Os KMs permitem criar um processo de integração que é modular, flexível e extensível. KMs são *templates* que definem o fluxo de dados e a geração de código.

Os KMs são genéricos já que permitem a geração de fluxos de dados, independentemente das regras de transformação escolhidas. Ao mesmo tempo, são muito específicos já que o código que geram e a estratégia de integração que implementam são otimizados para uma determinada tecnologia ou uma determinada plataforma.

O ODI possui um sistema de detecção de mudanças denominado de **CDC (Change Data Capture)** que permite detectar as mudanças num sistema operacional de forma a processar só os dados que foram actualizados e não todos os dados do sistema operacional. O ODI usa *triggers* no sistema operacional de forma a detectar qualquer operação DML (*insert*, *update*, *delete*) em certas tabelas escolhidas de antemão. Essas tabelas são denominadas de tabelas *journalizadas*. Os *triggers* perante qualquer operação DML inserem no *Journal* da tabela a chave primária do registo alterado, de forma a identificar o

registo alterado de forma única.

De modo a poder processar as mudanças de forma consistente, existe um processo de subscrição as actualizações que são feitas a um *journal*. Assim, é possível ter vários *subscribers* que detectam as mesmas mudanças na base de dados mas em intervalos de tempo diferentes ou para executar diferentes processos. Uma base de dados relacional é um modelo com muitas relações entre tabelas, pelo que uma alteração numa tabela significa provavelmente modificações em diversas tabelas. De forma a garantir a coerência e consistência das mudanças, existe um modo de CDC, o *Consistent Set*, que permite garantir que se processam conjuntos de actualizações, num determinado intervalo de tempo, em que a consistência dos dados é garantida.

A arquitectura do ODI é organizada à volta de um repositório central, que é acedido num modo cliente-servidor pelos diferentes módulos gráficos e pelos agentes de execução escritos em Java. A Figura 2.8 representa essa arquitectura de forma simplificada.

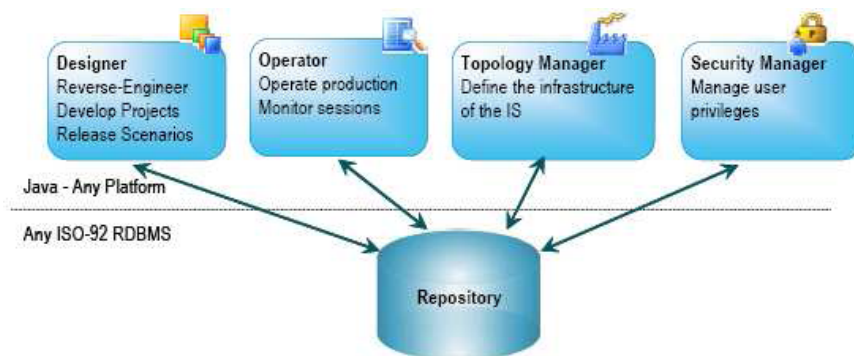


Figura 2.8: Arquitectura da aplicação ODI.

Existem 4 módulos gráficos do ODI que permitem editar de forma gráfica diversas operações de ETL.

O primeiro e o mais importante módulo gráfico é o **Designer**. O designer permite definir regras de transformações de dados e de integridade. Basicamente, qualquer desenvolvimento sobre o processo ETL é efectuado no Designer. É no designer que os metadados da base de dados são importados e definidos. Os metadados e as regras criadas permitem gerar cenários para produção. Este é o módulo principal para os programadores e os administradores da metadata.

O **Operator** permite gerir e monitorizar o ambiente de produção. Está destinado a pessoas envolvidas no ambiente produção e mostra logs de execução com contagens de erros assim como o número de linhas processadas, estatísticas de execução, o código gerado e executado, entre outros. O operator também pode ser usado durante a fase de desenvolvimento como ferramenta de debug.

O **Topology Manager** permite definir a arquitectura lógica e física da infra-estrutura. Servidores, schemas de base de dados e agentes são registados no repositório mestre

através deste módulo, geralmente pelos administradores da infra-estrutura ou do projecto.

O último módulo gráfico é o **Security Manager**. Este módulo permite gerir os perfis de utilizadores e os seus privilégios de acesso. O módulo também permite determinar privilégios de acessos a objectos e funcionalidades.

### 2.5.6 SVN



Subversion (também conhecido por SVN, o nome da sua ferramenta de linha de comando) é um sistema de controlo de versão desenhado especificamente para ser um substituto moderno do CVS, que se considera ter alguns defeitos. Um dos clientes mais populares de SVN é o Tortoise SVN, aplicação cliente gratuita para sistemas operativos Windows. A utilização do SVN, ou de forma geral de um sistema de controlo de versões, permite:

- **Controle do histórico:** Facilidade em analisar o histórico do desenvolvimento, como também facilidade na recuperação de versões mais antigas e estáveis. A maioria das implementações permitem analisar as alterações com detalhes, desde a primeira versão até a última.
- **Colaboração:** um sistema de controlo de versão permite que diversas pessoas trabalhem sobre o mesmo conjunto de documentos ao mesmo tempo e minimiza o desgaste provocado por problemas com conflitos de edições. É possível que a implementação tenha um controlo sofisticado de acesso para cada utilizador ou grupo de utilizadores.
- **Marcação de versões estáveis:** A maioria dos sistemas permite marcar onde é que o documento estava com uma versão estável, podendo ser facilmente recuperado no futuro.
- **Ramificação de projecto:** A maioria das implementações permite a divisão do projecto em várias linhas de desenvolvimento, que podem ser trabalhadas paralelamente, sem que uma interfira na outra.

## Capítulo 3

# Análise do problema

### 3.1 Contextualização

Este Projecto incide na definição e implementação de mecanismos que permitam a extracção de dados de um sistema de informação, a sua transformação, e carregamento noutro sistema de informação. Concretamente, o sistema fonte neste projecto é a aplicação ALERT® PRIVATE PRACTICE enquanto que o sistema alvo é o ALERT® DATA WAREHOUSE PRIVATE PRACTICE, ou ADW PP. De forma a perceber qual a melhor forma de efectuar o carregamento dos dados de uma aplicação para a outra, uma análise das características de cada uma das aplicações é necessária.

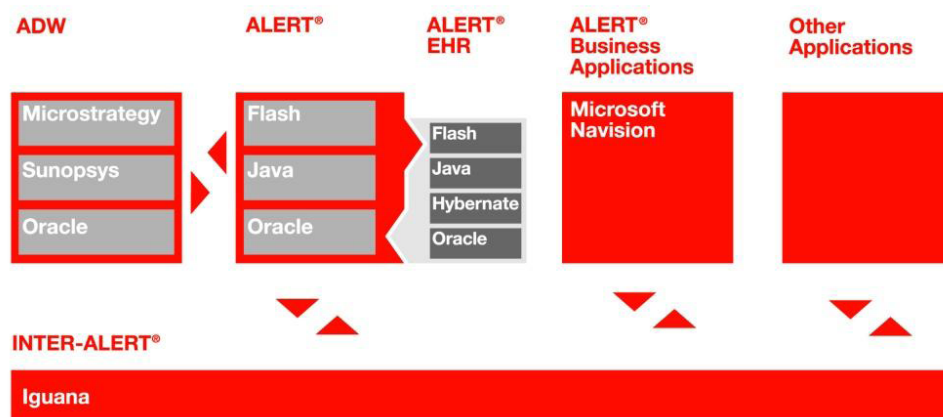


Figura 3.1: Tecnologias usadas no desenvolvimento da família de produtos ALERT.

### 3.1.1 Sistema fonte

O ALERT® PRIVATE PRACTICE faz parte da família de produtos ALERT, sendo um software destinado à informatização de clínicas e consultórios médicos. Como todos os produtos ALERT®, destina-se a todos os profissionais de saúde de uma instituição e permite registar todo o processo clínico que seria normalmente registado em papel. As principais funcionalidades do produto podem ser divididas nas seguintes áreas:

- Configuração de perfil de utilizador (administrador, médico, enfermeiro) dividido em diversas categorias
  - Edição da língua e dados biométricos (usados para autenticação no sistema);
  - Métodos de introdução de informação (livre ou categorizada);
  - Terminar ou configurar fim de turno;
  - Configuração pessoal da apresentação de dados.
- Listagem dos pacientes das seguintes formas:
  - Grelha das consultas do profissional;
  - Grelha das consultas da instituição.
  - Agenda.
- Informação clínica do paciente que inclui a visualização e edição de:
  - História clínica;
  - Doença actual;
  - Meios Complementares de Diagnósticos e Terapêuticas (MCDTs), por exemplo análises e exames;
  - Avaliação física;
  - Medicação;
  - Cuidados de enfermagem;
  - Alta e receita médica.

Na Figura 3.2 é apresentado um exemplo da interface da aplicação ALERT® PP (PRIVATE PRACTICE).

O ALERT® é desenvolvido em várias tecnologias, como se pode verificar na Figura 3.1, das quais se destacam as três principais:

- Oracle, na Base de Dados;
- Java, na camada intermédia;

## Análise do problema

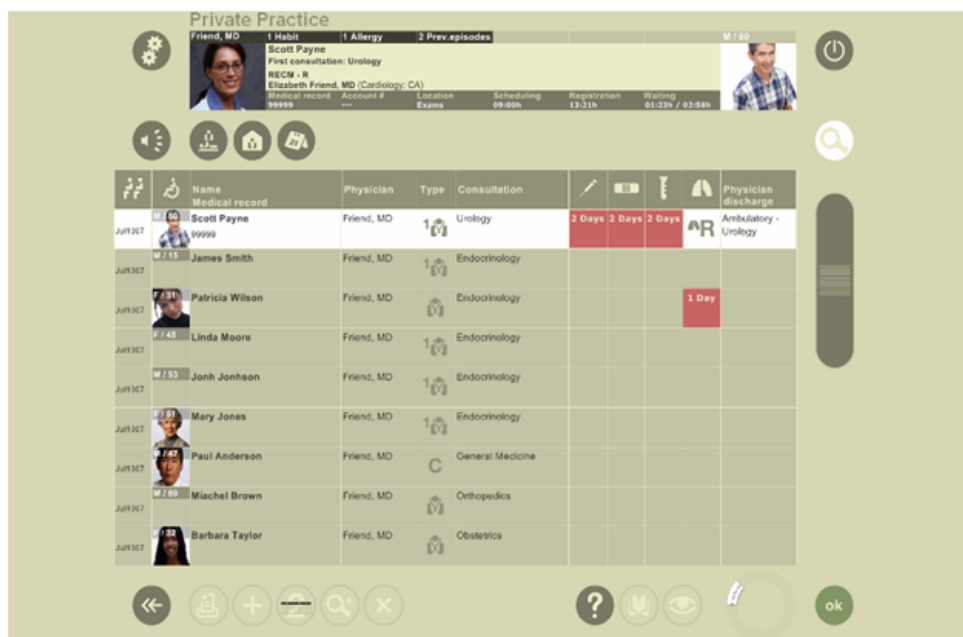


Figura 3.2: Interface do ALERT PRIVATE PRACTICE.

- Flash, na interface com o utilizador.

O ALERT® PP é um sistema transaccional e permite registar processos de negócio, pelo que as operações ao nível de dados são:

- Inserção, ou registo, de dados.
- Actualização da dados.
- Apresentação da informação.

Outro elemento importante de analisar é a forma como são armazenados os dados na aplicação. Como foi visto, a base de dados utilizada é Oracle e o modelo de dados é relacional e normalizado, que é um aspecto fundamental no sentido de analisar métodos de extracção de dados. Um sistema operacional tem como objectivos principais permitir múltiplas e rápidas transacções sobre um determinado modelo de dados. Um modelo de dados normalizado facilita as operações de inserção ou actualização, pois só é preciso actualizar a base de dados num sítio.

### 3.1.2 Sistema de destino

O sistema alvo do nosso carregamento de dados é o ADW PP. O ADW faz parte das soluções ALERT® e destina-se ao arquivo e análise de informação clínica e operacional, permitindo a realização de pesquisas, análises e relatórios complexos. O acesso à

informação é controlado de um modo eficaz. Através da utilização da informação contida no ADW, o utilizador é capaz de detectar tendências e identificar padrões, tornando-o num suporte valioso para a interpretação exacta dos eventos que decorrem no interior de um ambiente clínico. A informação apresentada nos relatórios do ADW resulta da utilização total ou parcial da informação obtida a partir das aplicações ALERT®. O ADW contém gráficos e relatórios padronizados e permite que os utilizadores criem os seus próprios relatórios. É uma fonte de informação adaptável e durável que foi concebida para alterações contínuas. As principais funcionalidades do ADW são:

- Apresentar de dados sob forma de gráficos ou tabelas;
- Produzir relatórios predefinidos ou personalizáveis;
- Manipular a informação apresentada:
  - Filtrar;
  - Fazer drill (descer/subir a um nível mais/menos detalhado dos dados apresentados);
  - "Pivotar" os dados (transformar colunas para linhas e vice-versa);
  - Criar métricas derivadas;
  - Ordenar dados;
  - Formatar dados;
- Possibilidade de "subscriver" relatórios;
- Ter vários perfis de utilizadores com diferentes níveis de visualização dos dados.

O ADW distingue-se pelas seguintes vantagens:

- Disponibilizar a informação para consulta em "tempo quase real";
- Analisar os dados sob várias vertentes (temporal, geográfico, demográfico, entre outras);
- Valioso instrumento de auxílio na tomada de decisão, com base em informação real.

A arquitectura da aplicação ADW pode ser dividida em três componentes principais como pode ser comprovado na Figura 3.1. A Interface do utilizador é construída com a aplicação MicroStrategy e o armazenamento de dados é feito num Data Warehouse Oracle. A integração dos dados da base de dados do ALERT para o *data warehouse* (ADW) é feito através de procedimentos PL/SQL e da ferramenta ODI (Oracle Data Integrator).

Na Figura 3.3 é apresentado um exemplo da interface da aplicação.

## Análise do problema

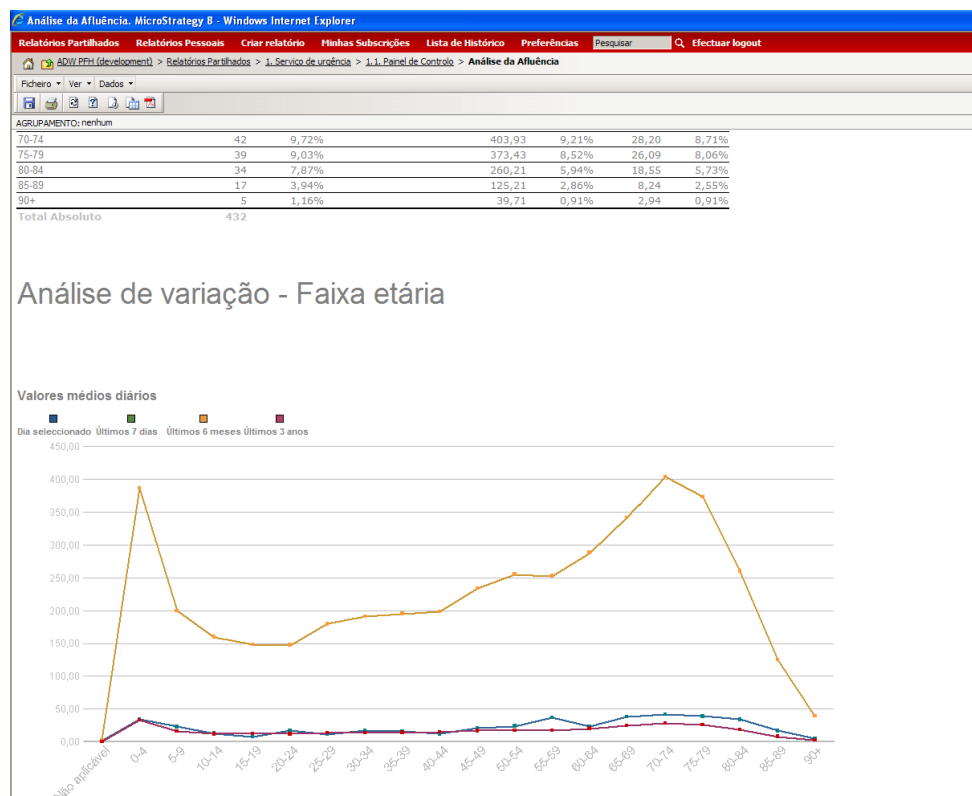


Figura 3.3: Interface da aplicação ADW.

Ao contrário do sistema fonte, o ADW é uma aplicação para avaliar processos de negócio pelo que a manipulação de dados envolve principalmente apresentação de informação com queries ad-hoc, isto é, interrogações que são imprevisíveis assim como interrogações padrões. Os principais objectivos de um sistema destes são:

- Compreensão fácil para o utilizador;
- Bom desempenho nas queries feitas;
- Capacidade de alteração;
- Coerência e consistência da informação.

O modelo de dados usado neste sistema é uma base de dados relacional que assenta num modelo dimensional e num desenho em estrela. O modelo de base de dados do *data warehouse* está, consequentemente, dividido em tabelas de factos e dimensões que representam a informação contida no ALERT.

### 3.1.3 Objectivos

O principal problema que se coloca neste projecto é a construção de um processo ETL entre o sistema ALERT e o ADW. Além disso, o modelo de dados do destino também deverá ser definido. Uma visão geral do projecto é representada na Figura 3.4.

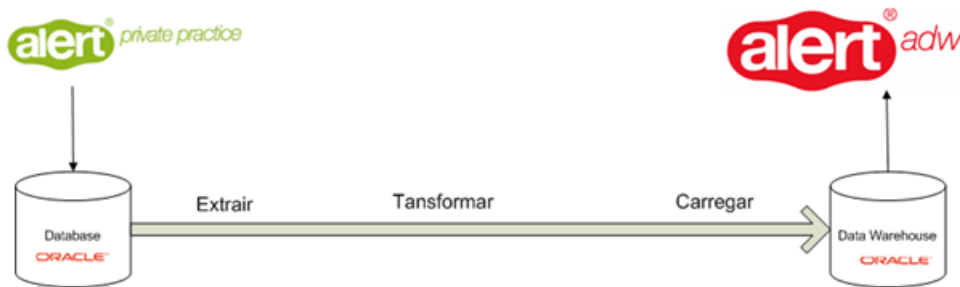


Figura 3.4: Visão geral do projecto.

Os principais objectivos e tarefas para a concretização do projecto devem passar pelos seguintes problemas:

- Quais os indicadores financeiros, clínicos e administrativos necessários?
- Quais são os melhoramentos que podem ser feitos aos dados actualmente carregados para se adaptar ao contexto de um *data warehouse* para consultórios privados?
- Quais são as novas fontes de dados, na base de dados de origem, a incluir no carregamento do *data warehouse*?
- Quais são as fontes de dados a manter?
- Que tecnologias e ferramentas devem ser usadas na extracção e carregamento de dados?
- Que método escolher para a extracção e carregamento de dados?
- Qual é o formato do modelo de dados mais adequado para as novas fontes de dados?
- O formato de dados do destino dos dados actualmente carregados está adaptado a este projecto?

No final do projecto, é esperado obter um mecanismo que permita o carregamento dos dados da base de dados do ALERT® PRIVATE PRACTICE para a base de dados do ADW. Estes dados deverão estar num formato adequado para posterior pesquisa pela ferramenta de visualização destes dados. Os dados escolhidos deverão também ser de grande relevância para o contexto de negócio do projecto, as clínicas e os consultórios privados. O plano de trabalho para o desenvolvimento do projecto é apresentado a seguir:

1. Fase de Análise:

- Análise dos indicadores existentes no actual *data warehouse*;
- Análise dos indicadores necessários ao nível de gestão e ao nível clínico para o *data warehouse*;
- Análise do sistema fonte;

2. Fase de Conceptualização

- Definição do modelo de dados de destino;
- Definição dos dados a extrair da origem;
- Definição do processo de carregamento;
- Integração das ferramentas e tecnologias a usar;

3. Fase de Implementação

- Criação do modelo de dados;
- Implementação do processo de carregamento;
- Definição de casos de testes e invariantes;

4. Fase de Testes

- Execução dos testes unitários;
- Controlo de qualidade dos dados;
- Controlo de eficiência do processo.

Um diagrama de Gantt que representa no tempo este planeamento encontra-se no anexo [A](#).

## 3.2 Situação actual do ADW

Uma análise do sistema de destino é essencial por forma a perceber que tipo de integração é necessário efectuar e como efectua-la.

### 3.2.1 Arquitectura do ADW

De forma a analisar o problema em causa é necessário analisar os pressupostos, isto é, analisar a arquitectura da aplicação na situação actual. A arquitectura da aplicação é representada na Figura [3.5](#) e pode ser claramente separada em duas áreas:

## Análise do problema

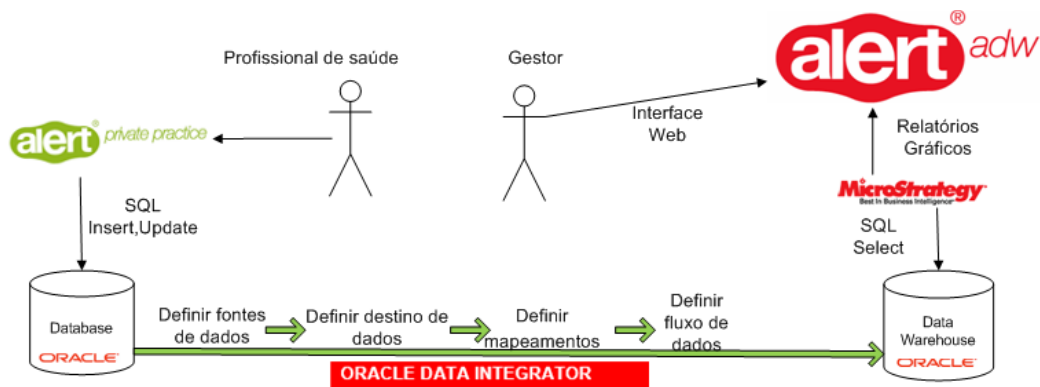


Figura 3.5: Arquitectura da aplicação ADW.

- área de Base de Dados (BD) e ETL;
- área de interface do utilizador.

O desenvolvimento de BD/ETL implica a criação do modelo de dados do *data warehouse* e do carregamento do mesmo por um processo ETL. Para efeitos de desenvolvimento do processo ETL usa-se a ferramenta Oracle Data Integrator que permite, através de uma interface gráfica, criar um processo ETL de forma completa, sendo ele responsável por gerar o código e automatizar o processo, com a possibilidade de criar agendamentos de execuções. Para mais informações sobre o Oracle Data Integrator, ver a Secção 2.5.5. O processo ETL também pode conter partes directamente escritas em SQL ou PL/SQL. O desenvolvimento do projecto está incluído nesta área.

O desenvolvimento da interface do utilizador implica a disponibilização de informação proveniente do ADW, num formato conveniente para visualização. As principais tarefas nesta área incluem a criação e visualização de relatórios predefinidos e de documentos personalizáveis. A aplicação MicroStrategy permite esses desenvolvimentos e a sua apresentação via interface Web. Essa interface é fácil de usar e acessível através de um *browser*. Para mais informações sobre o MicroStrategy ver a Secção 2.5.4. Os desenvolvimentos da camada da interface do utilizador, sobre esta plataforma, estão fora do âmbito deste projecto.

Apesar de existirem várias aplicações ALERT® com requisitos de negócios diferentes, dirigidas para diferentes tipos de cuidados de saúde (urgência, consulta externa, consulta privada, entre outros) existe um só modelo de dados para todos estes produtos. Deste modo, é possível, por exemplo, integrar vários produtos na mesma instituição de forma mais eficiente. A camada de interface com o utilizador e as regras de negócio é que variam de produto para produto, mas usam o mesmo modelo de dados. Alguns produtos usam só algumas tabelas mas são instalados e desenvolvidos sobre o mesmo modelo de dados global. Da mesma forma, e se existem variantes do produto ADW que são paralelos

aos vários produtos do ALERT®, existe um só *data warehouse* (um só modelo de dados) para todas essas variantes. Esse aspecto é fundamental para efeitos de desenvolvimento. A Figura 3.6 representa este conceito que envolve a utilização de um só modelo de dados para a aplicação.

A aplicação MicroStrategy acede ao mesmo modelo de dados mas trata a informação de forma diferente de produto para produto. São criadas métricas, relatórios, documentos específicos para cada produto a partir do MicroStrategy e é assim possível simplificar o desenvolvimento. O desenvolvimento do ETL e da interface do utilizador incidem sobre um só modelo de dados para todos os produtos.

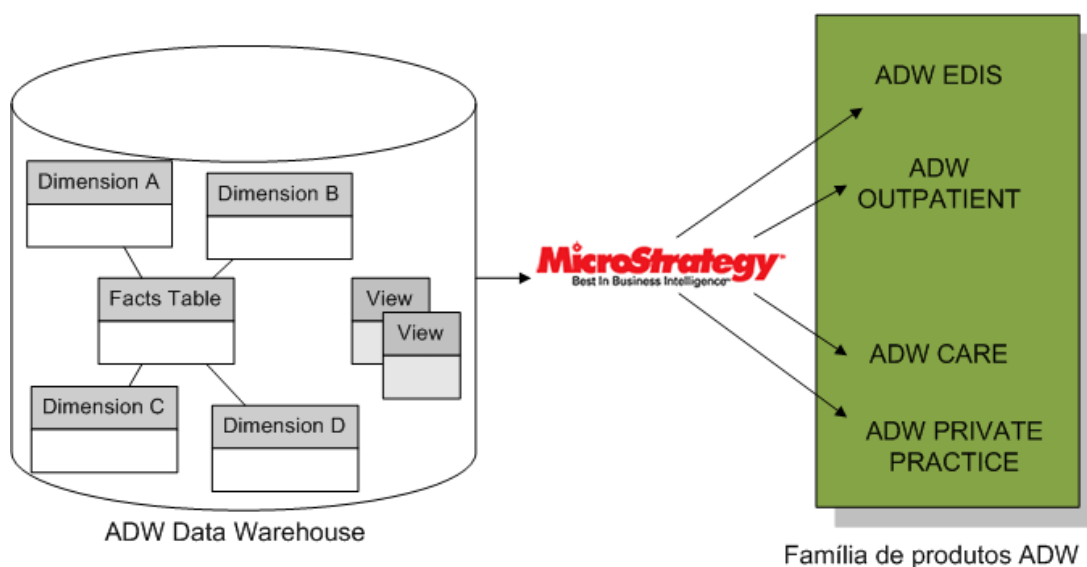


Figura 3.6: ADW - Um modelo de dados, vários produtos.

### 3.2.2 Processo ETL do ADW

O modelo de dados do ADW obedece aos princípios do modelo dimensional e está dividido em tabelas de factos e dimensões. As principais tabelas de factos do ADW são relativos aos:

- Episódios;
- Diagnósticos;
- Análises;
- Exames;
- Tarefas.

As principais dimensões do ADW são relativas ao:

- Tempo;
- Geografia;
- Paciente;
- Profissional;
- Demografia;
- Instituição.

As dimensões da geografia e da demografia fazem parte dos atributos comuns do ADW. A estrutura dos atributos comuns é específica do ADW e foi criada no intuito de maximizar a eficiência do processo ETL. Nos atributos comuns guardam-se os identificadores de todas as dimensões transversais a qualquer estrela de forma a efectuar o cruzamento entre essas dimensões e tabelas de factos de uma só vez. As dimensões que fazem parte dos atributos comuns são dimensões conformes, isto é significam o mesmo em qualquer estrela. De forma a serem sempre carregados em qualquer processo, os atributos comuns estão associados a um episódio. O episódio, em regra geral, está associado a qualquer processo dentro de uma instituição de saúde, pelo modelo de dados da ALERT®. Consequentemente, a arquitectura actual de carregamento de dados do ADW está centrada no conceito de episódio, como no modelo de dados das aplicações ALERT® em que todos os conceitos giram à volta de um episódio. Aquando da inserção ou actualização de um registo, a primeira operação feita pelo processo ETL é determinar a que episódio corresponde esse registo de forma a actualizar os atributos comuns e depois inserir ou actualizar efectivamente o registo.

O Oracle Data Integrator (ODI) serve de base para a implementação do processo ETL do ADW. Os processos ETL no ODI são definidos em *Packages*. Num *package* são definidos diferentes passos de execução e a sequência lógica de execução. Os passos podem ser procedimentos a executar na base de dados (SQL e PL/SQL), comandos do sistema operativo, interfaces, avaliação/atribuição de variáveis, entre outros. Um exemplo de criação de um *package* é apresentado na Figura 3.7. A partir dos *packages* é possível gerar cenários que podem ser agendados para correr em determinados intervalos de tempo.

## Análise do problema

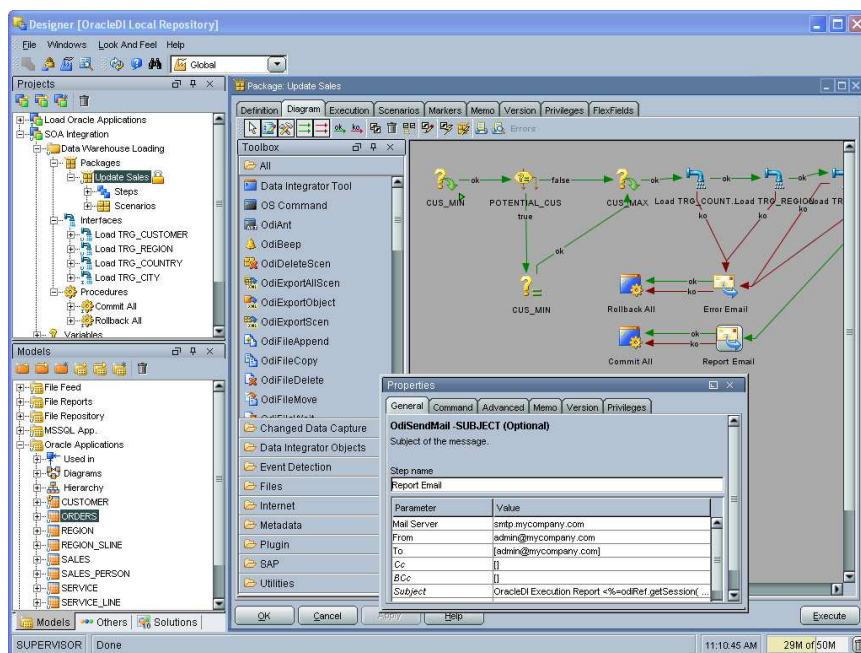


Figura 3.7: Exemplo de criação de um package no ODI.

Uma interface é o passo em que se definem transformações e mapeamentos de dados. A partir de uma interface gráfica é possível definir a tabela de destino e as tabelas fontes. Relativamente às tabelas (ou vistas) fontes pode-se criar todo o tipo de cruzamento e de filtros. Na tabela de destino pode-se definir os mapeamentos e transformações pretendidas (em formato SQL ou nativo da BD). Um exemplo de criação de uma interface é apresentado na Figura 3.8.

O processo ETL do ADW é um *package* geral que contém vários passos, alguns dos quais definidos em outros *packages*. Uma visão geral do ambiente do ADW que inclui o processo ETL, está representado de forma simplificada na Figura 3.9.

A criação do ambiente é o primeiro passo a efectuar e é necessário para o funcionamento do processo ETL. Esse ambiente consiste na estrutura de detecção de actualizações nas tabelas fonte do sistema operacional. A detecção de actualizações, ou "journalização" no ODI, segue um sistema de *publisher-subscriber* que está representado de forma simplificada na Figura 3.10.

É possível subscrever as mudanças no sistema operacional. Se uma tabela fizer parte do *CDC*, isto é, se fizer parte das tabelas que vão ser monitorizadas (journalizadas), então, no momento de criação do ambiente, é criada uma estrutura, o *journal* da tabela, que guarda as chaves primárias dos registos que foram alterados na tabela fonte. Além de criar os *journals*, o passo de criação de ambiente ainda cria os *triggers* na base de dados do sistema operacional. Esses *triggers* inserem nos *journals* os registos alterados após qualquer operação DML de *insert*, *update* ou *delete*. Quando estiverem criados todos os *journals*, *triggers* e a estrutura das tabelas internas do ODI para processar os *subscribers*,

## Análise do problema

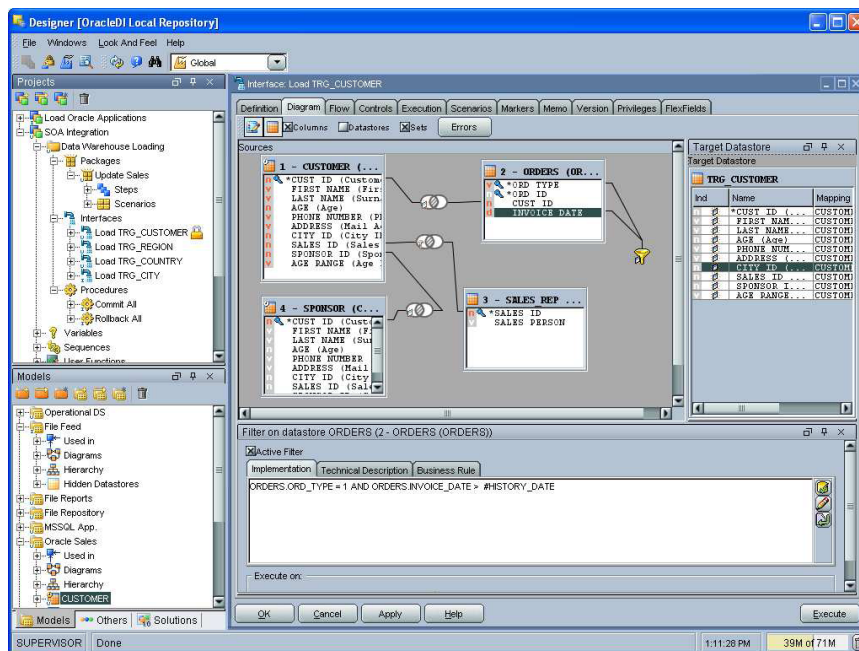


Figura 3.8: Exemplo de criação de uma interface no ODI.

o processo de ”journalização” é iniciado e o ambiente está preparado para o processo ETL.

Depois de estar criado o ambiente que permite processar as alterações no sistema operacional, o processo ETL pode ser executado. Geralmente, é agendado para correr de 10 em 10 minutos de forma a ter dados actualizados quase em *real time*.

Na Figura 3.9 é possível ver que o primeiro passo do processo ETL envolve a extensão da janela de CDC e o bloqueio dos *subscribers*. Estas operações são necessárias para processar as actualizações que ocorreram numa determinada janela (intervalo de tempo) e para um determinado *subscriber*. Desta forma obtém-se dados consistentes e que só envolvem um determinado período de tempo. Esta precaução é necessária devido ao facto de existir a possibilidade de ocorrerem operações de DML durante o processo ETL. As vistas sobre os *journals*, ou vistas de ”journalização”, são criadas de forma a aceder aos dados alterados nas tabelas ”journalizadas” para um determinado *subscriber*, num determinado intervalo de tempo.

De seguida, um procedimento é executado no sistema operacional de forma a processar todos os *journals* e construir uma tabela que contém todos os episódios alterados, assim como o processo de negócio (Estrela) a que está associada essa modificação. Desta forma cada estrela só processa os processos de negócio alterados nesses episódios. Por exemplo a estrela das análises, só irá processar análises efectuadas ou alteradas desde o último processamento.

O próximo passo no processo envolve o processamento de cada dimensão. Esse processamento é efectuado de forma paralela com um *package* por dimensão. Cada um desses *packages*, tipicamente, detecta se houve actualizações no sistema operacional para

## Análise do problema



Figura 3.9: Processo ETL do ADW.

a dimensão em causa e se houve alterações é actualizada a dimensão. Se não houve alterações o processo passa directamente ao passo seguinte.

Depois de terem sido processadas todas as dimensões, é necessário actualizar a tabela dos *atributos comuns*. Essa tabela contém registos com o identificador de um episódio e as chaves anónimas das dimensões dos atributos comuns correspondentes. Esta tabela é carregada no *data warehouse* antes de carregar as tabelas de factos. Desta forma, o cruzamento com as dimensões atributos comuns nunca é feito na própria estrela. É sempre feito a partir dessa tabela, o que permite maximizar a eficiência.

Quando o passo anterior estiver concluído, é possível, então, carregar as tabelas de factos de todas as estrelas. Assim como acontece com as dimensões, cada tabela de factos é carregada num *package*, e esses *packages* são executados de forma sequencial. Cada *package* verifica, primeiro, se houve alterações para aquele processo de negócio e, se houver, processa de seguida as alterações para aquele processo de negócio e insere a informação correspondente numa tabela de *staging*. Essa tabela de *staging* é cruzada com os atributos comuns e as dimensões de forma a carregar a tabela de factos. Quando um registo a inserir não respeitar alguma regra de integridade, esse registo é considerado erro e é inserido numa tabela de erros. Cada tabela de factos tem a sua tabela de erros, de forma a guardar de forma persistente os registos que não foram carregados num determinado momento devido a um erro.

No fim do processamento de cada uma das tabelas de factos, chegámos ao último passo do processo ETL. É necessário desbloquear os *subscribers* e limpar os *journals*. Só são apagados os registos do *journal* se todos os *subscribers* que existem já tiverem consumido aqueles dados.

Quando o processo ETL já não precisa de correr ou quando queremos alterar de forma substancial o CDC, é necessário destruir o ambiente, isto é destruir a estrutura de

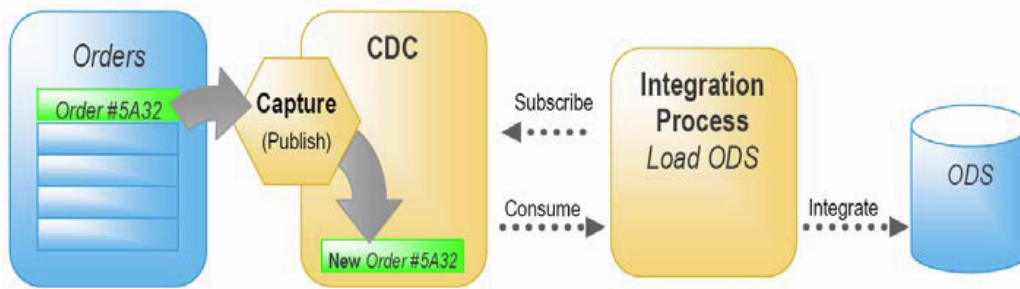


Figura 3.10: Modelo de detecção de actualizações no ODI.

”journalização” do ambiente ETL. Para esse efeito, o primeiro passo consiste na remoção dos *subscribers*, enquanto que o segundo consiste na paragem do processo de ”journalização”. Essa paragem envolve a destruição dos *triggers* e tabelas que foram criados no sistema operacional durante a criação do ambiente.

### 3.2.3 Indicadores existentes

A família de produtos ADW oferece já uma vasta gama de indicadores possíveis sobre cuidados de saúde. Indicadores são medidas ou avaliações que podem ser feitas sobre um determinado negócio. Neste contexto de negócio, podem-se dividir os tipos de indicadores disponíveis em duas categorias:

- Indicadores clínicos;
- Indicadores de gestão administrativa.

De forma a avaliar as necessidades de novos desenvolvimentos para o novo produto ADW PRIVATE PRACTICE, é necessário também avaliar o que já é possível disponibilizar com os desenvolvimentos já efectuados. Na tabela 3.1 são apresentadas as principais estrelas do *data warehouse* assim como o tipo de indicadores que existem actualmente.

Actualmente, grande parte dos indicadores existentes são indicadores clínicos. Esta circunstância deve-se principalmente ao facto, como já foi descrito anteriormente, dos produtos ALERT® serem orientados sobretudo ao processo clínico que decorre na instituição de saúde. A maior parte das instituições possui, alias, *software* administrativo e financeiro. Desta forma, os indicadores administrativos são reduzidos e os indicadores financeiros são inexistentes, devido ao facto do próprio produto ALERT® não registar essa informação. Essas limitações de dados do próprio sistema operacional são fundamentais pois um *data warehouse* só contém informação que existe no sistema fonte.

Tabela 3.1: Informação existente no ADW.

| <b>Estrela</b>    | <b>Descrição</b>                                                                       | <b>Exemplo de indicadores</b>                                                       |
|-------------------|----------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|
| Episódios         | Um episódio corresponde à passagem de um paciente por uma instituição de saúde         | data da primeira observação; número de regressos em 15 dias; número de re-aberturas |
| Diagnósticos      | Diagnósticos médicos registados na aplicação. Existem diagnósticos diferentes e finais | Data de registo; Data de confirmação                                                |
| Exames            | Requisição de exames (imagem ou outros)                                                | Data da requisição; Data do resultado                                               |
| Análises          | Requisição de análises                                                                 | Data da requisição; Data da colheita                                                |
| Transferências    | Pedidos de transferências de instituição e de responsabilidade                         | Data da requisição; Data de execução                                                |
| Medicação interna | Prescrições internas de medicamentos, soros, vacinas                                   | Dosagem; intervalo entre tomas; número de tomas                                     |
| Prescrições       | Prescrições de medicamentos para o exterior (receita médica)                           | Data de impressão; Quantidade prescrita                                             |
| Problemas         | Problemas dos pacientes (Problemas, alergias, hábitos)                                 | Data de registo do problema; Data de início do problema                             |
| Tarefas           | Tarefas dos profissionais por episódio, hora, instituição                              | Número de requisições de análises; Número de inserções de diagnósticos              |

### 3.3 Necessidades para o ADW PRIVATE PRACTICE

O ADW PRIVATE PRACTICE tem como fonte de dados o ALERT® PRIVATE PRACTICE, um produto destinado às clínicas privadas e consultórios. Este produto tem as suas especificidades e as suas semelhanças relativamente a outro tipo de produtos ALERT® destinados para instituições de saúde com características diferentes. Algumas das principais características a ter em conta são:

- Agenda de consultas;
- Implementação no mercado americano iminente;
- Conceito de co-sign.

Um dos aspectos importantes do ALERT® PRIVATE PRACTICE relativamente aos outros produtos, é o facto de estar centrado em agendamentos de consultas. Efectivamente, a única maneira de criar um novo episódio (passagem de um paciente pela instituição de saúde e por consequente pelo ALERT) é pelo meio de uma consulta. As consultas podem ser marcadas para uma determinada hora tanto por um médico como por um administrativo da instituição.

Na estrela dos episódios, existem alguns factos relativos ao agendamento associado ao episódio. Contudo, o conceito de agendamento implica que a um determinado momento pode existir um agendamento sem que exista um episódio associado, pois a hora

da consulta pode ser posterior à data actual. Essa situação ainda não é contemplada pelo *data warehouse*, já que como foi visto qualquer dado inserido actualmente no ADW tem um episódio associado que cruza informação com as dimensões denominadas de *atributos comuns*. Como é óbvio, a informação sobre os agendamentos que ainda não têm episódios associados ou que nunca chegaram a ter, por terem sido cancelados, tem um valor significativo numa clínica privada, pelo que a sua informação deverá ser carregada de algum modo para o *data warehouse*.

Actualmente, o ADW PRIVATE PRACTICE está em fase de implementação iminente no mercado americano, pelo que é necessário analisar as necessidades para esse mercado relativamente ao *data warehouse*. Um dos aspectos principais está relacionado com a língua. Efectivamente, é necessário ter a capacidade de guardar e disponibilizar os dados na língua de origem, neste caso o inglês.

Além do problema da língua, um novo mercado implica novas regras de negócio e novos conceitos. Um desses conceitos é conceito de co-sign. O Co-sign é muito utilizado nas clínicas privadas nos Estados Unidos e por isso torna-se fundamental, ao nível de informação clínica, disponibilizar informação sobre ele. Nos Estados Unidos, existem outros tipos de profissionais de saúde do que em Portugal. Uma dessas categorias é o *Physician assistant* que não tem o nível de educação de um médico mas que tem bases sólidas de medicina. Esse tipo de profissional atende muitos pacientes em clínicas mas as suas decisões clínicas devem ser sempre validadas por um médico. Essa validação é intitulada de *co-sign*. Os *Physician assistant* fazem pedidos de co-sign e os médicos podem validar ou não o co-sign, validando assim uma decisão clínica. Associado a este conceito de co-sign está o conceito de sign-off que está associado com o fecho de um episódio. Erros clínicos graves podem surgir com problemas de co-sign pelo que essa informação é vital no *data warehouse*.

O ADW PRIVATE PRACTICE é um produto que tem como mercado alvo os gestores das clínicas privadas ou consultórios. Sendo assim, estes utilizadores querem uma visão geral da situação da clínica. Os indicadores administrativos e financeiros são, deste modo, uma grande mais-valia. Os indicadores mais relevantes neste contexto são indicadores relacionados com:

- Avaliação de desempenho dos profissionais;
- Avaliação de comportamento/satisfação dos clientes;
- Informação financeira.

O desempenho dos profissionais numa clínica privada tem de ser controlado, de forma a manter um nível de qualidade rigoroso no atendimento ao cliente e de forma a controlar

as despesas. Efectivamente, uma clínica precisa de gerir e controlar as tarefas efectuadas pelos profissionais de saúde, e mecanismos que permitem avaliá-los de forma isenta devem ser encontrados.

Um dos aspectos fundamentais para uma clínica privada é a avaliação do comportamento dos clientes. Efectivamente, ao contrário de um hospital ou de um centro de saúde, o atendimento dos clientes numa clínica privada é essencial. Os gestores querem ver o tipo de comportamento dos pacientes, querem comparar essa informação e cruzá-la de diferentes formas usando indicadores específicos de uma clínica privada, como a consulta e o tipo de consulta. Desta forma, alguns dos indicadores que foram avaliados como importantes foram indicadores tais como: o tempo que um cliente demorou a ser atendido, que tipo de agendamentos de consultas costumam atrasar-se mais, entre outros. Essa informação ainda não existe no ADW mas está relacionado com a inclusão da informação sobre agendamento de consultas que já foi referida.

Outro aspecto fundamental para um clínica privada é o financeiro. Efectivamente, apesar de prestar cuidados de saúde, uma clínica privada tem de ser rentável e uma das formas de verificação é avaliar os indicadores que estão disponíveis. Sobretudo num mercado como os Estados Unidos, os indicadores financeiros são essenciais para destacar um produto de *data warehouse*.

No entanto, o ALERT® é um produto com vertente clínica pelo que a informação financeira não existe no *software*. A inclusão de informação de outros sistemas operacionais sem ser o ALERT® também não é um processo fácil. Pois, o próprio cliente tem de consentir na partilha de informação doutros sistemas. Além disso, a ALERT está a desenvolver produtos administrativos para as instituições pelo que a inclusão de outros sistemas operacionais não faz sentido neste contexto, de momento.

A maior parte dos indicadores clínicos do ADW são adaptados ao contexto do ADW PRIVATE PRACTICE, nomeadamente a informação sobre análises, exames, episódios, diagnósticos e tarefas deverá ser incluída no novo produto. Os indicadores que podem ser utilizados estão listados na tabela 3.1.

### 3.4 Limitações

Apesar de existirem indicadores interessantes no modelo actual do ADW, existem também algumas limitações. A principal limitação está relacionada com a estrela das tarefas dos profissionais. Nomeadamente, os principais problemas identificados são:

- Estrela das Tarefas com problemas de desempenho;
- Estrela das Tarefas não inclui toda a informação necessária sobre as tarefas efectuadas durante um episódio;

- Impossibilidade de fazer o *tracking* da passagem do paciente pela instituição de saúde;
- Impossibilidade de ver agregações diárias, mensais, anuais de forma eficiente e directa.

Os problemas de desempenho da estrela das tarefas estão directamente relacionados com o processo ETL. Efectivamente, todos os dados são carregados a partir do modelo de dados do ALERT®. Estamos a falar de uma quantidade considerável de dados, já que haverá sempre várias tarefas por episódio. Por isso, é uma das estrelas com maior quantidade de informação a processar, no modelo fonte, e a armazenar, no *data warehouse*. Mas, se actualmente os dados são provenientes do ALERT®, é necessário analisar se estes dados não podem ser carregados a partir das estrelas do ADW. Efectivamente, a estrela das tarefas contém informação por exemplo sobre análises requisitadas, exames requisitados, diagnósticos registados, entre outros. No entanto, essa informação já existe no *data warehouse*, noutro formato, nas diversas estrelas. Pelo menos parte da informação tem, portanto, a possibilidade de ser carregado a partir do *data warehouse*, e assim evitar o sobrecarregamento do sistema operacional.

Além da carga excessiva sobre o sistema operacional, a quantidade de dados a processar (cruzamentos entre grande parte das tabelas do sistema operacional) faz do processo de carregamento dessa estrela o mais pesado de todo o processo. Na Figura 3.11 apresenta-se o exemplo de execução do processo ETL em 3 hospitais distintos. É notória a duração excessiva do processo de carregamento das tarefas relativamente ao processo ETL total. Efectivamente, quase metade do tempo do processo ETL é gasto a carregar as tarefas dos profissionais. Além desse facto, no próprio processo de carregamento das tarefas mais de metade do tempo de carregamento é devido ao carregamento dos dados do sistema operacional (ALERT®) para uma área de *staging* (ainda na base de dados do ALERT®).

Outro aspecto limitativo na estrela das tarefas está directamente relacionado com o nível de detalhe da informação armazenada. Efectivamente, a granularidade da estrela impede a disponibilização de alguns dados. A estrela está agregada:

- à hora;
- ao episódio;
- ao profissional;
- à instituição;

Esta agregação impede, por exemplo, de saber se duas tarefas ocorreram na mesma hora qual é que ocorreu primeiro. Da mesma forma, se for necessário saber a data exacta

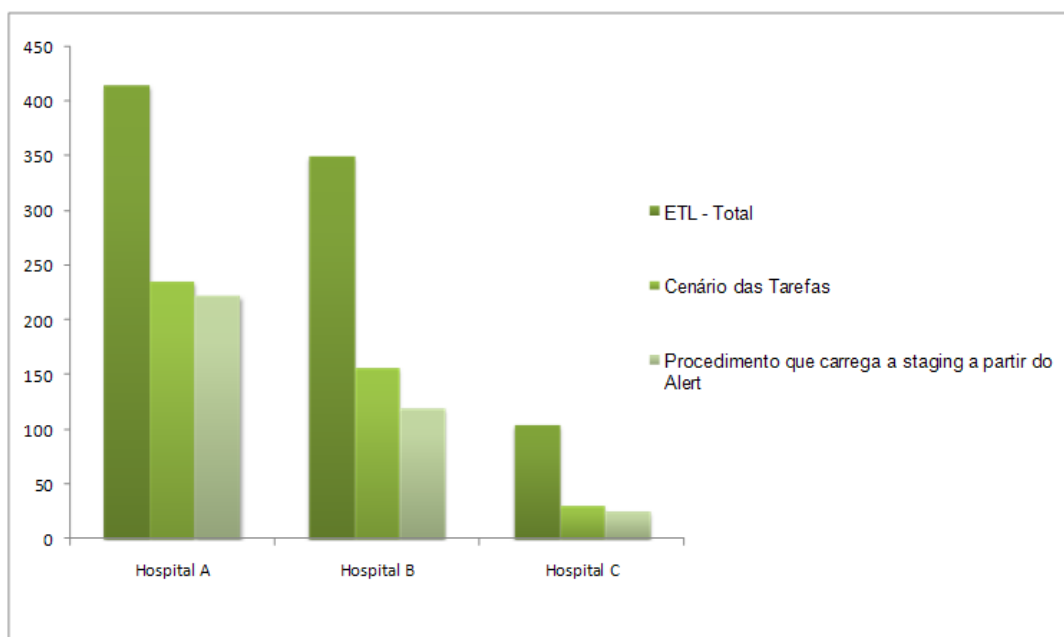


Figura 3.11: Duração (em segundos) do processo de carregamento da estrela dos profissionais.

de execução de uma tarefa é impossível com este modelo. Num caso mais prático, por exemplo, é impossível visualizar todas as tarefas associadas a um paciente, durante um episódio, de forma cronológica. Esse tipo de informação pode ser muito importante no caso de despiste de problemas ou erros no atendimento a um paciente. Um gestor de uma instituição pode também querer saber qual a primeira tarefa a ser mais vezes executada pelos profissionais.

Além disso, a estrela está actualmente agregada à hora mas este tipo de informações tem grande valor com dados sobre outros períodos de tempo nomeadamente dias, meses e anos. Como já foi referido, a estrela das tarefas é a estrela que contém a maior quantidade de dados do *data warehouse* pelo que operações de agregação sobre essa estrela, no momento de interrogações, podem ser muito pesadas em termos de recursos e evidenciar problemas de performance. Torna-se, consequentemente, difícil visualizar, de forma fácil e em tempo real, informações diárias, mensais e anuais sobre as tarefas dos profissionais.

### 3.5 Resumo ou conclusões

A arquitectura actual do ADW, e os desenvolvimentos já efectuados sobre ele no âmbito de outros produtos, permite utilizar a maior parte dos processos já implementados. Desta forma, todas as estrelas já existentes no modelo de dados podem servir de base para o ADW PRIVATE PRACTICE.

No entanto, existem algumas especificidades que necessitam de novos desenvolvimentos. Nomeadamente, a inclusão do conceito de agendamentos, a adaptação dos dados

geográficos para o contexto norte-americano e a adição de alguns indicadores específicos são essenciais.

Além disso, algumas das funcionalidades actuais não satisfazem totalmente as necessidades reais. O caso da estrela das tarefas dos profissionais é o caso mais grave e prioritário. Efectivamente, os indicadores das tarefas são uns dos indicadores mais importantes do ADW pelo facto de serem objectivos, fáceis de visualizar e comparar. Contudo, existe uma limitação na granularidade da estrela que impede ver mais informação. Além disso, a performance do processo de carregamento dessa estrela é muito baixa, criando até problemas de performance no próprio ALERT® devido a acessos excessivos à base de dados. Desta forma, parece de extrema importância uma reformulação dessa estrela de forma a melhorar o produto final.

## Capítulo 4

# Modelo de dados do ADW PRIVATE PRACTICE

Na secção anterior, chegámos à conclusão que eram necessários novos desenvolvimentos, em termos de modelo de dados e processo ETL, de forma a satisfazer as necessidades do novo produto ADW PRIVATE PRACTICE.

Nomeadamente, estes novos desenvolvimentos podem subdividir-se em três fases de desenvolvimento:

- Nova estrela de agendamentos;
- Reformulação da estrela das tarefas;
- Adaptações do modelo actual.

### 4.1 Estrela dos agendamentos

O novo conceito de agendamento corresponde, claramente, a um novo processo de negócio. Tem uma granularidade diferente de qualquer outra estrela e inclui dados que não existem actualmente no modelo de dados do ADW. Logo, detectámos a necessidade da construção de um novo novo data mart (estrela). Seguindo o modelo de Kimball para a construção de um *data warehouse* (ver secção 2.3), já encontramos o nosso novo processo de negócio que é o primeiro passo. Falta-nos agora, definir a granularidade da nova tabela de factos, as dimensões e os factos.

Como já vimos, o conceito de agendamento destaca-se, principalmente, pelo facto de poder haver agendamentos sem existir episódio no ALERT. Esse aspecto tem de ser

levado em consideração na escolha da granularidade da tabela de factos. A definição da granularidade da tabela de factos dos agendamentos deverá ser então:

*Um registo da tabela de factos dos agendamentos corresponde à marcação de uma consulta ou doutro tipo de evento, denominado por agendamento, por um profissional para um paciente para um determinado horário numa determinada instituição e para uma determinada especialidade.*

Assim, cada registo da tabela de factos da estrela dos agendamentos corresponde à marcação de um evento. A cada marcação de um evento estão associados diversos atributos que precisam de ser identificados, separados e categorizados de forma a acrescentá-los de forma correcta como factos ou atributos das dimensões. A escolha das dimensões é o próximo passo, na construção da nova estrela do *data warehouse*.

Como foi referido anteriormente a estrela dos episódios contém no modelo actual informação sobre o agendamento associado a um episódio. Com esta reformulação, estes factos da estrela dos episódios não fazem sentido permanecer. No entanto, as dimensões criadas para o efeito podem ser utilizadas para a nova estrela. As dimensões em causa são:

- **Tipo de agendamento:** Esta dimensão inclui informação sobre o tipo de agendamento, nomeadamente se é um agendamento para uma primeira consulta ou para uma consulta subsequente;
- **Razão de cancelamento do agendamento:** Esta dimensão inclui informação sobre o motivo de cancelamento do agendamento (por exemplo: um engano, falta do médico, falta do utente);
- **Tipo de evento agendado:** Esta dimensão inclui informação sobre o tipo de evento da consulta associada ao agendamento, como por exemplo, consulta médica, consulta de especialidade, consulta de grupo;
- **Meios de marcação do agendamento:** Esta dimensão inclui informação sobre o meio de comunicação usado para marcar o agendamento (por exemplo: normal ou telefone);
- **Iniciativa do agendamento:** Esta dimensão inclui informação sobre a entidade que tomou a iniciativa do agendamento (por exemplo: o médico, o utente, ou a própria clínica);

Estas dimensões deverão ser incluídas na nova estrela. Deste modo, não se irão cruzar com a tabela de factos dos episódios mas, exclusivamente, com a tabela de factos dos agendamentos.

Além das dimensões referidas, um agendamento cruza-se com as dimensões *atributos comuns*. Esse cruzamento no caso de agendamentos que tenham um episódio associado é directo. No entanto, para os agendamentos sem episódios, o cruzamento deverá ser feito doutra forma pois o cruzamento das tabelas pelo episódio esta descartado. (ver Capítulo 5).

Apesar das dimensões citadas garantirem muita informação sobre os agendamentos, ainda não é suficiente. Efectivamente, no sistema operacional, existe informação sobre a categoria do agendamento e sobre o tipo de vagas de um agendamento. Uma categoria de agendamento pode ser por exemplo uma consulta, uma análise, um exame, ou ainda uma operação cirúrgica. Cada um destes conceitos consiste num novo tipo de informação que não se relaciona directamente com a informação existente nas dimensões e que por isso devem ser guardadas em novas dimensões. Os tipos de vagas de agendamentos podem ser normal, urgente, ou ainda além-vaga (em que uma vaga pode ser agendada para vários agendamentos) Assim, para responder ao conceito de categoria de agendamento, é necessário criar uma nova dimensão de categoria de agendamentos e para responder ao conceito de tipo de vaga de agendamento, é necessário criar uma nova dimensão de tipo de vaga de agendamentos.

As dimensões referidas anteriormente partilham o mesmo tipo de estrutura. Essa estrutura está descrita na tabela 4.1. As dimensões do ADW seguem a metodologia de tipo 2 (ver secção 2.2) para gerir as actualizações de dados nas dimensões.

Tabela 4.1: Estrutura das dimensões específicas dos agendamentos.

| <b>Campo</b>             | <b>Descrição</b>                                                                       | <b>Tipo de comportamento SCD</b> |
|--------------------------|----------------------------------------------------------------------------------------|----------------------------------|
| Chave anónima            | Identificador anónimo do registo da dimensão                                           | Surrogate Key                    |
| Data de carregamento     | Data do último carregamento/actualização                                               | Starting Timestamp               |
| Identificador do registo | Identificador do registo proveniente do sistema operacional                            | Natural Key                      |
| Código do registo        | Código que permite identificar de forma única o registo na dimensão                    | Overwrite on change              |
| Descrição do registo     | Descrição textual do registo que contém a informação a usar na interface do utilizador | Overwrite on change              |
| Data de início           | Data de início de disponibilização do registo                                          | Overwrite on change              |
| Data de fim              | Data de fim de disponibilização do registo                                             | Ending Timestamp                 |
| Estado do registo        | Flag que indica se o registo está activo ou é histórico                                | Current Record Flag              |
| Identificador da língua  | Identificador da língua para efeitos de cruzamento com os factos                       | Natural Key                      |

Os campos do tipo *Natural Key* são identificadores únicos de cada registo. Se houver um novo registo para inserção com identificadores únicos que já existem no ADW é necessário guardar o actual como histórico e inserir o novo registo. O novo registo terá no campo do tipo *Current Record Flag* um valor igual a 1 e no histórico um valor igual à 0. Os campos *Starting Timestamp* e *Ending Timestamp* correspondem, respectivamente, aos campos de início e fim do intervalo de tempo em que o registo esteve activo. Os campos *Overwrite on Change* são os que podem mudar numa actualização de um registo.

Finalmente, o campo correspondente à *Surrogate Key* é a chave anónima e primária, incrementada aquando da inserção/actualização de um novo registo.

Além das novas dimensões, é necessário escolher os factos que vão integrar a nova estrela. Alguns dos indicadores relativos aos agendamentos eram guardados na estrela dos episódios e podem agora ser incluídos na nova tabela. Os principais factos da estrela dos agendamentos são:

- Data de requisição, data de marcação e data de cancelamento do agendamento;
- Data prevista de início e fim do evento.
- Duração prevista.

A estrela dos agendamentos está representada de forma simplificada no diagrama da Figura 4.1.

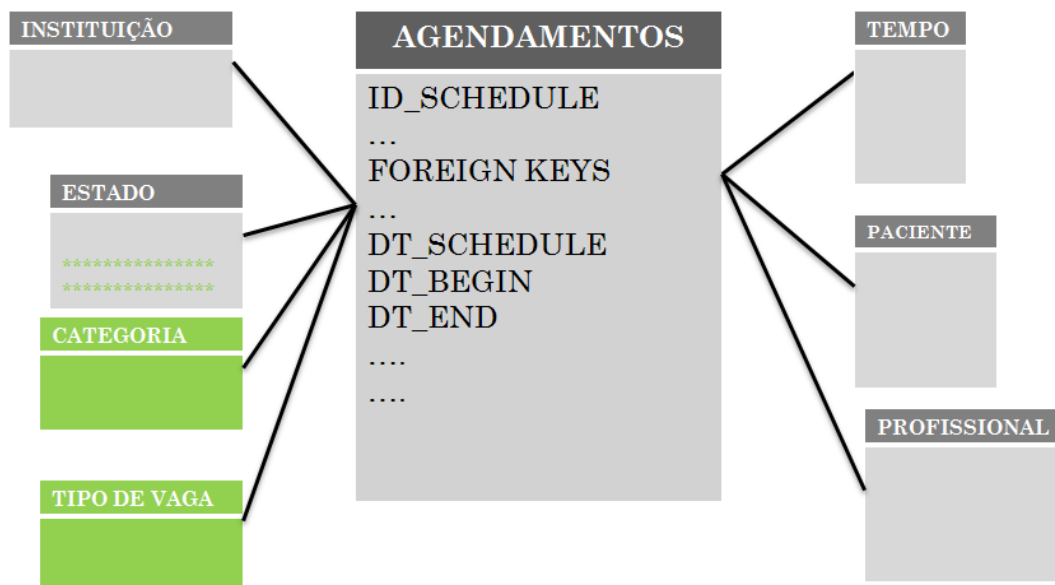


Figura 4.1: Estrela dos agendamentos.

## 4.2 Estrela das tarefas dos profissionais

As tarefas dos profissionais já estão representadas no ADW sob a forma de uma estrela. No entanto, a informação presente não é a mais completa e o processo ETL sofre de problemas de performance e de peso excessivo sobre a base de dados do ALERT. Neste sentido, é necessário reformular tanto o modelo de dados como o processo ETL. O modelo de dados actual da estrela das tarefas está representado na Figura 4.2.

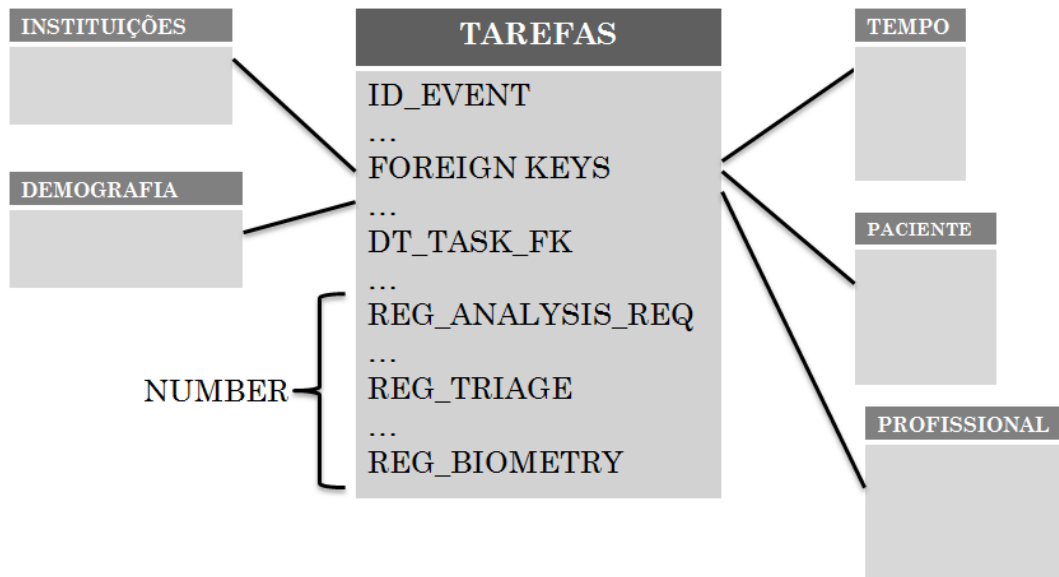


Figura 4.2: Estrela das tarefas dos profissionais - modelo anterior.

Neste modelo existem tantos factos como tipo de tarefas, com dados agregados à hora, ao episódio, ao profissional, ao paciente e à instituição. Mas este tipo de agregação não nos permite obter a data de execução da tarefa apesar dessa informação existir no sistema operacional. A granularidade actual pode ser definida pela seguinte frase:

*Cada linha da tabela de tarefas corresponde ao registo de tarefas efectuadas a um paciente, por um determinado profissional, numa determinada instituição, num determinado episódio, durante uma hora.*

Para resolver o problema, existem várias alternativas possíveis que devem ser analisadas.

#### 4.2.1 Alternativa 1

A primeira alternativa é uma estrela em que se registam todas as tarefas e respectivas datas, usando uma dimensão para identificar o tipo de tarefa. De forma a garantir o acesso fácil à informação seriam também criadas tabelas de agregação à hora, dia e mês. Essa alternativa é representada na Figura 4.3.

Se analisarmos o impacto do novo modelo, podemos concluir, facilmente, que reduzimos de forma drástica o número de colunas na tabela de factos pois já não temos um coluna por tarefas. Temos agora uma coluna que representa o tipo de tarefa, que substitui as 70 colunas do modelo anterior correspondentes à 70 tipos de tarefas diferentes. No entanto, a tabela de factos passava a ter muito mais linhas, devido ao facto de cada tarefa passar a corresponder a um registo. Se tentarmos relacionar o número de tarefas e o número de episódios chegámos a conclusão que:

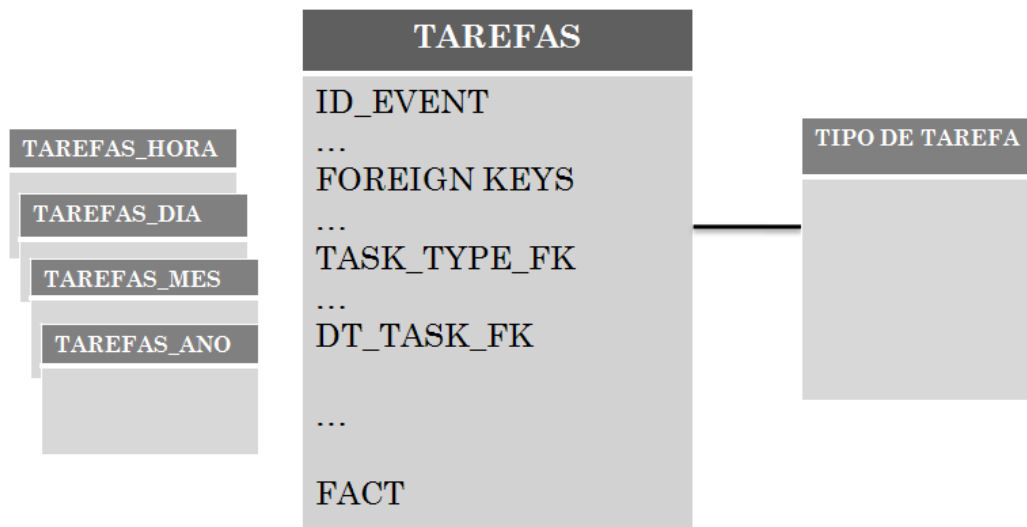


Figura 4.3: Estrela das tarefas dos profissionais - primeira alternativa.

$$NumMedTable = NumMedEpisodes \times NumMedTasksPerEpisode \quad (4.1)$$

**NumMedTable** Número médio de linhas da tabela de factos das tarefas

**NumMedEpisodes** Número médio de episódios

**NumMedTasksPerEpisode** Número médio de tarefas por episódio

Uma análise das tendências em diversas instituições mostra que em média o número de tarefas situa-se à volta dos 35. Sendo que esse número não varia substancialmente de instituição para instituição, podemos analisar a tendência média de crescimento da tabela de factos das tarefas. A tabela 4.2 compara o crescimento da tabela de factos das tarefas com o número de episódios. Foi considerado que cada registo teria um tamanho médio de 150 bytes de forma a avaliar o espaço em disco.

Tabela 4.2: Crescimento das tarefas em relação ao número de episódios.

| Episódios | Tarefas    | Espaço em disco (MegaBytes) |
|-----------|------------|-----------------------------|
| 1.000     | 35.000     | 5                           |
| 10.000    | 350.000    | 49                          |
| 100.000   | 3.500.000  | 488                         |
| 1.000.000 | 35.000.000 | 4.874                       |

O crescimento da tabela torna-se um problema complexo a partir de um milhão de episódios, situação em que ultrapassamos os 4 Gigas de espaço em disco só para a tabela. Na prática, o número de episódios ainda não atinge esse valor mas esse crescimento de

espaço, por questões de performance, tem de ser controlado. Algum tipo de particionamento (dividir em várias partes no intuito de arquivar informação) no fim de alguns anos de utilização do ALERT® e ADW pode ser necessário de forma a sustentar esse modelo.

Apesar do problema da quantidade de informação armazenada, esta solução oferece diversas vantagens:

- Informação detalhada e completa sobre todas as tarefas;
- Cada registo da tabela ocupa menos espaço em disco;
- responde a qualquer tipo de interrogação à base de dados.

#### 4.2.2 Alternativa 2

Outra possibilidade considerada é a divisão da tabela das tarefas em várias tabelas, de forma a optimizar as operações de DML. A divisão poderia ser feita agrupando categorias de tarefas semelhantes. Teríamos tabelas de factos de tarefas do tipo *Análises* ou *Exames*, entre outros. A Figura 4.4 representa essa alternativa. A vantagem dessa alternativa seria dividir a carga por várias tabelas ao nível de espaço ocupado e do processo ETL.

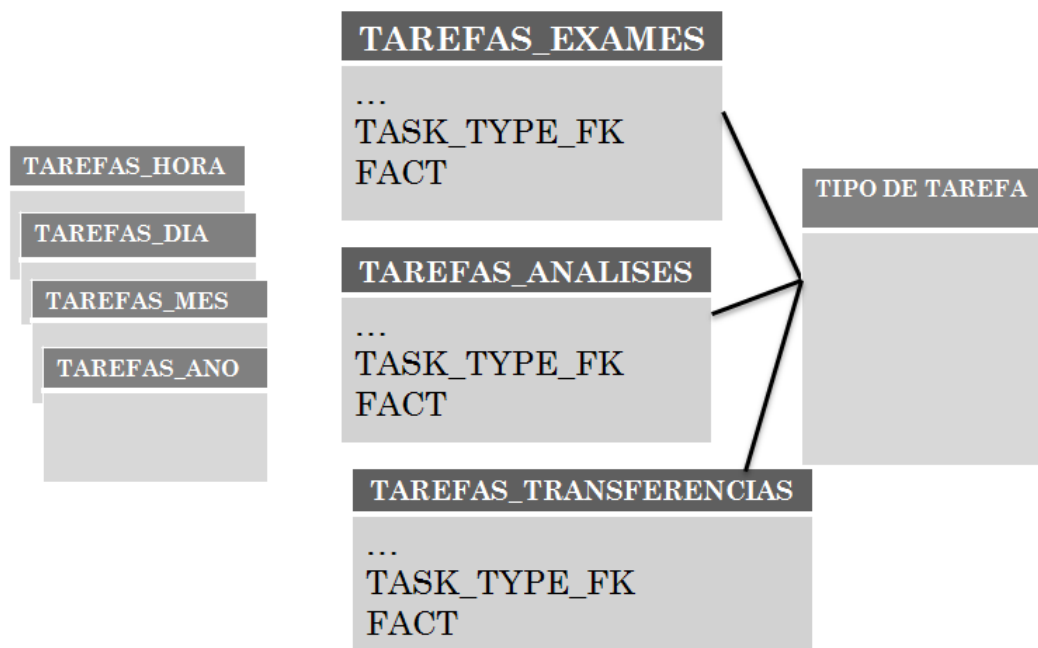


Figura 4.4: Estrela das tarefas dos profissionais - segunda alternativa.

#### 4.2.3 Alternativa 3

A terceira alternativa envolve guardar o modelo actual em termos de tabela de factos. No entanto, de forma a melhorar a performance do lado do utilizador seriam criadas

agregadas ao mês, dia e ano no mesmo tipo de estrutura que a tabela de facto. Esta solução tem a vantagem de não mexer num processo ETL funcional já que o carregamento da tabela de factos permaneceria idêntico. Contudo, esta alternativa tem a desvantagem de não poder mostrar uma tarefa ao nível mais baixo. A Figura 4.5 representa essa alternativa.

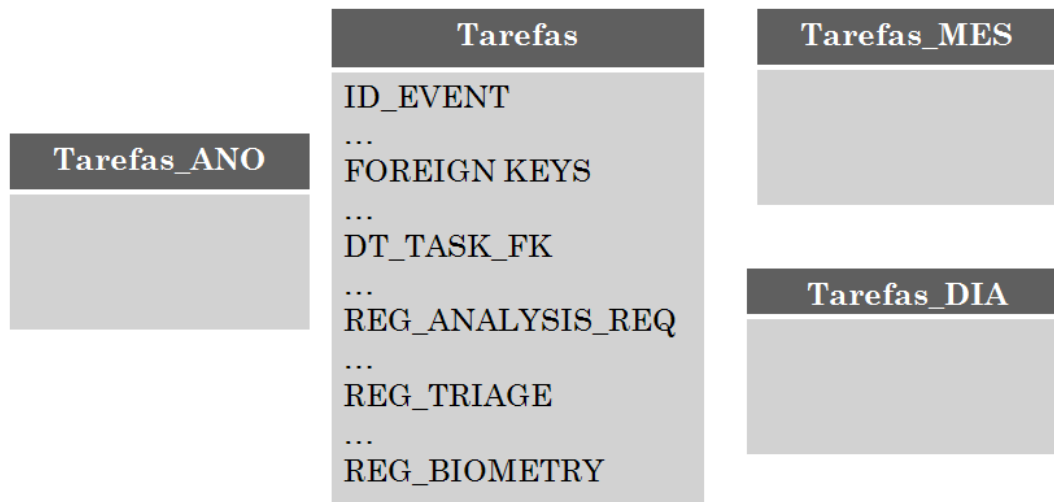


Figura 4.5: Estrela das tarefas dos profissionais - terceira alternativa.

#### 4.2.4 Alternativa 4

A quarta alternativa envolve o mesmo modelo de tabela de factos que a primeira e segunda alternativa. As agregadas seriam, no entanto, armazenadas na mesma estrutura que a tabela de factos. Essa alternativa tem a desvantagem de alterar o modelo para a interface do utilizador e pode prejudicar a eficiência da mesma. A vantagem seria um menor peso dessas agregadas, tanto em termos de construção como de espaço ocupado em disco. A Figura 4.6 representa essa alternativa.

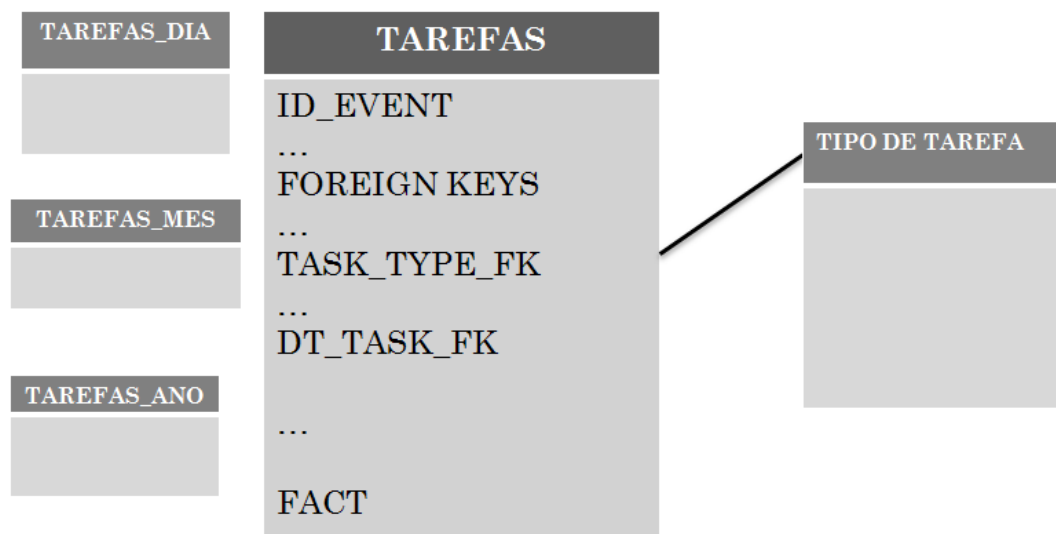


Figura 4.6: Estrela das tarefas dos profissionais - quarta alternativa.

#### 4.2.5 Definição do novo modelo

As grandes desvantagens das hipóteses 3 e 4 descartam-nas a partida como opções viáveis. Efectivamente, as suas limitações podem comprometer a interface do utilizador ou não compensam um desenvolvimento novo relativamente ao ganho realizado. Apesar de ser uma possibilidade interessante, a segunda alternativa, devido ao facto da descentralização da informação, representa uma complexidade que não compensa o ganho em termos de performance. Efectivamente, apesar de distribuímos as tarefas por diversas tabelas, a estrutura é idêntica em cada uma. Logo, acabámos por não ganhar nada em termos de espaço em disco. Além disso a optimização das operações DML acaba por ser muito relativa. Na prática, uma inserção de 10 registos numa tabela de 1 milhão de registos e uma inserção de 10 registos em 5 tabelas de 200.000 registos não difere muito, pelo que a primeira alternativa parece ser a mais viável.

De forma a definir a construção da nova estrela, seguindo a primeira alternativa, devemos definir a granularidade da tabela de factos. A granularidade da tabela deve definir de forma explícita o que cada registo representa. À partida, basta especificar que cada tarefa diferente do ALERT deve corresponder a um registo diferente na tabela de factos. No entanto, o conceito de tarefa é um conceito que só existe no ADW, pois as tarefas correspondem às diversas operações distintas realizadas, ao nível do sistema operacional. Desta forma, necessitamos de um mecanismo que permita identificar de forma única os registos provenientes do sistema operacional.

No sistema operacional, as tarefas provêm de diversas tabelas e podem existir várias tarefas distintas provenientes da mesma tabela. Desta forma, temos de ter em atenção

o facto de poder haver tarefas com o mesmo identificador mas que provêm de tabelas diferentes. Desta forma, o identificador da tabela do sistema operacional não nos permite definir de forma única uma tarefa. Além disso, como pode haver várias tarefas distintas guardadas na mesma tabela, não podemos identificar uma tarefa só pelo identificador e pela identificação da tabela do sistema operacional. É necessário saber também o tipo de tarefa, de forma a determinar a unicidade de um registo.

Assim, a chave única da tabela é portanto a combinação dos campos de:

- Identificador do sistema operacional;
- Identificação da tabela do sistema operacional;
- Identificação do tipo de tarefa.

A definição da granularidade da tabela pode ser definida pela seguinte frase:

*Cada linha da tabela de factos das tarefas corresponde à execução de um determinado tipo de tarefa à um determinado momento, por um determinado profissional, numa determinada instituição.*

De forma a responder aos novos requisitos da granularidade da tabela de factos das tarefas, é necessário a criação de uma nova dimensão: o tipo de tarefa.

A dimensão do tipo da tarefa tem os seguintes campos:

- Chave anónima;
- Data de carregamento do registo;
- Identificador do sistema operacional: de momento não existe mas pode vir a evoluir;
- Código das tarefa;
- Descrição da tarefa;
- Descrição do grupo ao qual pertence a tarefa;
- Código da tarefa de nível superior (da qual depende esta tarefa);
- Data de início de disponibilização do registo;
- Data de fim de disponibilização do registo;
- Estado do registo;
- Identificador da língua.

A dimensão do tipo da tarefa acaba por ser uma dimensão *especial*, no sentido que não contém nenhuma informação proveniente de forma directa do sistema operacional. Esta dimensão permite criar o conceito de tipo de tarefa e assim identificar os registos de execução de tarefas numa só tabela. Por não representar de forma directa a informação do sistema operacional, não faz sentido explicitar o comportamento SCD, já que as alterações à dimensão deverão ser feitas de forma manual.

Além desta dimensão, a estrela deve ter como dimensões: as atributos comuns, a instituição, o profissional e o tempo. Desta modo respeitamos a granularidade da tabela definida acima.

Cada registo da tabela de factos das tarefas representa a execução de uma tarefa pelo que o único facto disponível é a data da sua execução. Essa data é um *timestamp*, de forma a ter o momento de execução com uma precisão ao segundo.

Já foi referido a importância da visualização de dados sobre as tarefas dos profissionais sobre diversos períodos de tempo. As interrogações efectuadas sobre as tarefas impliquem na maior parte dos casos uma agregação temporal de forma a ver a informação em formato horário, diário ou mensal. Logo, faz sentido criar essas estruturas de forma persistente no próprio *data warehouse*, com o intuito de melhorar a performance do lado do utilizador do *data warehouse*. As agregadas sobre o tempo que fazem mais sentido são as agregadas à hora, dia e mês. As agregadas sobre períodos de tempo de maior duração, como por exemplo ao ano ou ao trimestre, não fazem sentido, já que o custo desses cálculos é muito reduzido. De forma a minimizar as alterações à *metadata* da interface do utilizador, as agregadas deverão seguir o modelo de dados anterior.

Desta forma, a agregada à hora mostra todas as tarefas agregadas à hora, episódio, profissional, paciente e instituição e segue o mesmo modelo de granularidade da antiga tabela de factos das tarefas descrita acima (ver Figura 4.2). Um novo facto que não existia foi acrescentado: a soma de todas as tarefas. Essa informação tem muita relevância para os utilizadores. Esse cálculo a nível de base de dados é bastante trivial pelo que deverá ser guardado de forma persistente como facto.

Para as agregadas ao dia e ao mês, a granularidade é ligeiramente diferente pois os requisitos para essas agregadas são diferentes. Neste caso, já não faz sentido distinguir a informação do episódio e dos pacientes. No entanto, é importante agregar sobre as dimensões dos atributos comuns, da triagem, do tipo de evento ou da causa externa, pela importância desses indicadores para uma visão temporal. Consequentemente, as agregadas ao dia e ao mês registam o número de tarefas ocorridas, durante um determinado dia ou mês, agregadas à instituição, profissional, tipo de evento, triagem e causa externa. Estas duas tabelas de agregação também têm dois novos factos: o número de eventos e o número de pacientes.

### 4.3 Adaptações do modelo actual

Um dos primeiros pontos a considerar é o carregamento correcto dos dados das dimensões, em várias línguas. Todas as dimensões contêm um campo que é um identificador da língua, que permite ter dados em diversas línguas para o mesmo registo dimensional. Sendo assim, dados guardados em línguas estrangeiras no sistema operacional podem ser carregados de forma correcta pois a estrutura das dimensões já está preparada para esse efeito.

Problemas podem surgir no caso em que a informação não provém sistema operacional. Este é o caso para as dimensões da geografia, tempo e demografia. Existem outras dimensões estáticas que não são carregadas a partir do sistema operacional, em que os seus dados são traduzidos de forma manual sem alterações estruturais.

No caso da geografia, a estrutura da dimensão está preparada para receber vários tipos de formatos de localização geográfica. A estrutura aceita, efectivamente, 5 níveis de descrição sobre uma localização geográfica. No mundo ocidental, a maior parte das moradas está associada a um código postal e na dimensão da geografia esse campo também existe. No caso de Portugal por exemplo, temos:

- Descritivo de nível 1 é o País;
- Descritivo de nível 2 é o Distrito;
- Descritivo de nível 3 é o Concelho;

A mesma estrutura é capaz de receber informações de localização geográfica para os Estados Unidos em que teríamos:

- Descritivo de nível 1 é o País;
- Descritivo de nível 2 é o Estado;
- Descritivo de nível 3 é o *County*.
- Descritivo de nível 4 é o Cidade;

No caso da dimensão do tempo, a estrutura também é flexível, contendo unicamente campos descritivos sobre o calendário gregoriano que devem ser traduzidos. Outros tipos de calendário, que não o gregoriano, não podem ser aplicados a esta estrutura pelo que pode implicar restrições à extensão a outros mercados. No entanto, o âmbito deste projecto é fornecer uma solução para o mercado nacional e dos Estados Unidos, pelo que uma dimensão baseada no calendário gregoriano é suficiente.

Outra adaptação está relacionada com a inclusão do conceito de co-sign no *data warehouse*. O co-sign inclui sobretudo informação relacionada com um episódio e tarefas

dos profissionais. Nomeadamente, podemos destacar os seguintes aspectos, que são os mais importantes a guardar no *data warehouse*:

- Pedidos de co-signs efectuados;
- Co-signs efectuados;
- Data do último sign-off;
- Data do último co-sign.

Os dois primeiros conceitos estão relacionados com tarefas efectuadas pelos profissionais, enquanto que os dois últimos estão relacionados com datas associadas a um determinado episódio. Assim, a forma mais fácil de acrescentar estes conceitos ao *data warehouse* é adicioná-las às estrelas já existentes pois estes conceitos não influenciam a granularidade das tabelas de factos já existentes.

Temos então na estrela das tarefas dois novos tipos de tarefas:

- Pedido de co-sign;
- Co-sign.

De forma paralela, na estrela dos episódios é necessário incluir dois novos factos:

- Data do último sign-off;
- Data do último co-sign.

De forma a incluir essa informação, o processo ETL tem de incluir essa informação nos dados a extrair (ver Capítulo 5).

Foi realçado no Capítulo anterior a importância de indicadores financeiros para um *data warehouse* de uma clínica privada. Também já foi referido a escassez deste tipo de indicadores, no sistema operacional ALERT. Contudo, existem dois campos presentes na base de dados do ALERT que podem ser de interesse para o ADW, nomeadamente:

- Preço da consulta;
- Flag que indica se o pagamento já foi efectuado ou não.

#### 4.4 Resumo ou conclusões

Neste Capítulo foram definidas as estruturas de dados necessárias para contemplar os novos desenvolvimentos do produto ADW PRIVATE PRACTICE. Essas novas estruturas integram-se no modelo de dados existente do *data warehouse*.

Para satisfazer os requisitos em termos de agendamentos de consultas, foi necessário definir uma nova estrela, que inclui uma tabela de factos de agendamentos e duas novas

dimensões: categoria e tipo de vaga do agendamento. Essa definição da nova estrela contém ainda a especificação da granularidade da tabela de factos, das dimensões a incluir na estrela e dos factos que virão a ser usado como indicadores.

Os problemas identificados com a estrela das tarefas dos profissionais, nomeadamente em termos de performance e nível de detalhe da informação, originaram a reformulação total da estrela das tarefas. A granularidade da tabela de factos foi alterada de forma a corresponder ao nível atómico de execução de uma tarefa. Para corresponder a esta granularidade, foi criada uma nova dimensão de tipo de tarefa. Os factos da nova tabela de factos foram, consequentemente, alterados de forma a incluir a data de execução de uma tarefa. Tabelas de agregação temporal foram especificadas, de forma a enriquecer o modelo de dados e permitir otimizar as interrogações mais frequentes dos utilizadores.

O modelo de dados existente teve de ser adaptado, de forma a responder as exigências do novo produto. Assim, novos factos foram adicionados à diversas estrelas e a inclusão de diversas línguas foi normalizada nas dimensões.

## Capítulo 5

# Processo ETL do ADW PRIVATE PRACTICE

Da mesma forma que foi essencial definir o modelo de dados para os novos desenvolvimentos identificados, é agora necessário definir o processo ETL, isto é o processo de extracção, transformação e carregamento dos dados para os seguintes casos:

- Estrela dos agendamentos;
- Estrela das tarefas dos profissionais;
- Adaptações do modelo actual.

A implementação de um processo ETL pode ser feito através de código (SQL e PL/SQL pro exemplo) ou através de ferramentas gráficas que automatizam o processo e permitem gerar o código. Uma dessas ferramentas é o Oracle Data Integrator usada durante a implementação do projecto. As secções [2.5.5](#) e a secção [3.2.2](#) descrevem, com mais detalhes, as funcionalidades e integração do Oracle Data Integrator no ADW.

### 5.1 Estrela dos agendamentos

#### 5.1.1 Desenvolvimento

O carregamento dos dados para a estrela dos agendamentos pode ser dividido em duas tarefas distintas: carregamento das dimensões e carregamento da tabela de factos.

O carregamento das dimensões do tipo SCD (Slowly Changing Dimension) é efectuado sempre no início do processo geral ETL. Esse carregamento é único e permite o carregamento de todas as estrelas do *data warehouse*. Desta forma, a maior parte das

dimensões usadas na estrela dos agendamentos já têm o processo ETL definido e implementado. Além do carregamento das dimensões do tipo SCD, é efectuado um carregamento inicial das dimensões estáticas. As dimensões estáticas são as dimensões que não se alteram ao longo do tempo. As dimensões estáticas são carregadas na implementação do processo ETL. Desta forma, só as novas dimensões especificamente criadas para a estrela dos agendamentos, precisam de ser implementadas no processo geral do ETL. Logo, surge a necessidade de definir e implementar o processo ETL para:

- A dimensão do tipo de vaga do agendamento;
- A dimensão da categoria do agendamento.

Apesar de serem dimensões que podem variar ao longo do tempo e que têm uma estrutura Slowly Changing Dimension, o método de carregamento é estático. Sendo carregadas como dimensões estáticas, o processo de carregamento resume-se às inserções em SQL incluídas num script a integrar com os outros scripts das dimensões estáticas. Esse script insere, não só os valores possíveis do sistema operacional, como outro valor especial: Registo de não aplicável.

Este registo é carregado em todas as dimensões e permite garantir a consistência da informação. Quando não é possível cruzar uma dimensão com uma tabela de factos porque um registo é *null*, o campo em causa da tabela assume o valor do registo do "Não aplicável" da dimensão. Da mesma forma, quando um cruzamento com uma dimensão falha porque o valor não existe na dimensão, o campo em causa da tabela de factos assume o valor do registo do "Não aplicável" da dimensão. A presença desse registo em todas as dimensões permite garantir a integridade e qualidade da informação.

Relativamente à tabela de factos, o processo ETL é mais complexo. Efectivamente, para carregar uma tabela de factos são necessários, geralmente, muitos cruzamentos da tabelas no sistema operacional, transformações avançadas de dados, e cruzamentos com dimensões.

O processo ETL geral do ADW é feito através de um *package* geral que contém vários *packages*. O carregamento de uma tabela de factos corresponde a um *package*. Na Figura 5.1 pode-se visualizar o *package* do processo ETL para a tabela de factos dos agendamentos.

O *package*, isto é o processo ETL da tabela de factos, está dividido em quatro passos:

- Actualização da variável dos agendamento;
- Avaliação do valor da variável dos agendamentos;
- Execução da interface da tabela de *staging*;
- Execução da interface da tabela de factos.

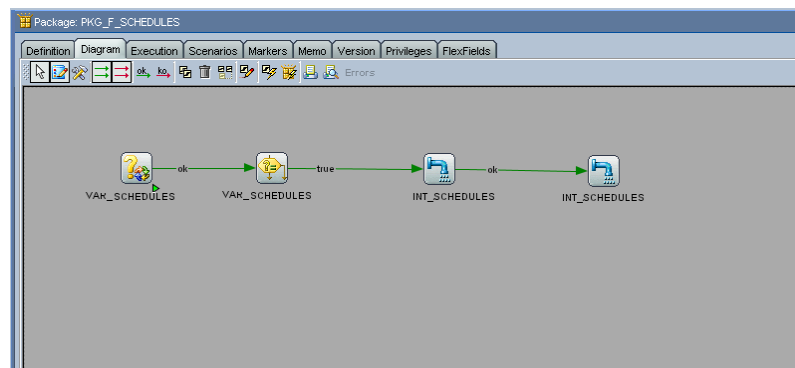


Figura 5.1: Visão geral do processo ETL dos agendamentos.

O primeiro passo corresponde à actualização da variável dos agendamentos. Esta variável representa o valor retornado por uma interrogação à base de dados, que conta os registos que foram actualizados desde o último processamento. Os registos que foram actualizados desde o último processamento podem ser contabilizados na vista de "journalização" sobre a tabela fonte dos agendamentos, no sistema operacional.

O segundo passo avalia o valor da variável dos agendamentos. Se for igual a zero, isto é se nenhum registo foi actualizado desde o último processamento, então o processo não é executado, pois não é preciso carregar nenhum registo já que nenhum foi inserido ou actualizado, no sistema operacional. Se a variável for superior a zero então o próximo passo é executado.

O terceiro passo consiste na execução da interface que carrega os dados do sistema fonte para a tabela de *staging* dos agendamentos. As tabelas de *staging* das tabelas dos factos são usadas no processo ETL como ponto de recuperação e para garantir a eficiência do processo. As tabelas de *staging* situam-se na base de dados do ALERT, permitindo, desta forma, fazer todos os cruzamentos necessários ainda na base de dados de origem, para maior eficiência. Cada interface no ODI tem um *knowledge module* (KM) associado de forma a definir o fluxo de dados. O KM nesta interface é o IKM Oracle Incremental Update. Este KM já é otimizado para base de dados Oracle e algumas das especificidades, em termos de fluxo de dados, são as seguintes:

- Faz um *update* se possível e se não for faz um *insert*;
- Sincroniza os *deletes* com as tabelas journalizadas incluídas na interface;
- Apaga todos os dados da tabela de destino (*staging*) antes da inserção;
- Apaga os objectos temporários criados pelo ODI para a criação das tabelas (pode ser desactivado para efeitos de debug);

- Efectua um *analyze* (permite otimizar a performance) à tabela de destino (*staging*).

Já que um agendamento pode não ter um episódio associado, o método tradicional de filtrar os registos a processar pelos episódios que foram alterados não pode ser usado, neste caso, no carregamento da *staging*. Desta forma, usa-se a vista de "journalização" da tabela dos agendamentos, no sistema operacional, para detectar que agendamentos foram alterados desde o último processamento.

Além disso, é necessário filtrar a informação que vem do sistema operacional. Uma das especificidades da base de dados do ALERT é nunca apagar dados mesmo quando estas operações são canceladas, por exemplo. Desta forma, este tipo de dados não devem ser carregados no *data warehouse* e devem ser filtrados o mais cedo possível. Logo, a filtragem tem de ser feita no carregamento da *staging*. Este controlo passa por aplicar os filtros desejados nas tabelas pretendidas na interface de carregamento da *staging*.

Na Figura 5.2 é possível ver, com mais detalhes, os passos na execução da interface de carregamento da *staging*:

1. Remover a tabela intermédia de integração, caso ela existe;
2. Criar a tabela de integração com uma estrutura semelhante à tabela de *staging*, com mais um campo que indica o tipo de registo (um *insert*, *update* ou *delete*);
3. Efectuar um truncate à tabela de destino, isto é são apagados todos os registos da tabela sem remover a própria tabela;
4. Inserir dados na tabela de integração a partir dos cruzamentos definidos na interface entre as tabelas do sistema operacional. Só os dados "journalizados" são carregados neste passo;
5. Sincronizar os registos apagados entre a vista de "journalização" e a tabela de integração e apagar esses dados na tabela de integração.
6. Criar na tabela de integração um índice sobre a chave única definida na tabela de destino, de forma a otimizar o processo.
7. Marcar todos os registos da tabela de integração que já existem na tabela de destino com uma flag de *update*.
8. Carregar a tabela de *staging*: os valores que já existem (com flag de *update* na tabela de integração) são actualizados e os que não existem (flag de *insert* na tabela de integração) são inseridos.
9. Fazer *commit* à transacção em curso, isto é as mudanças efectuadas são validadas de forma persistente fora da sessão de execução do processo.

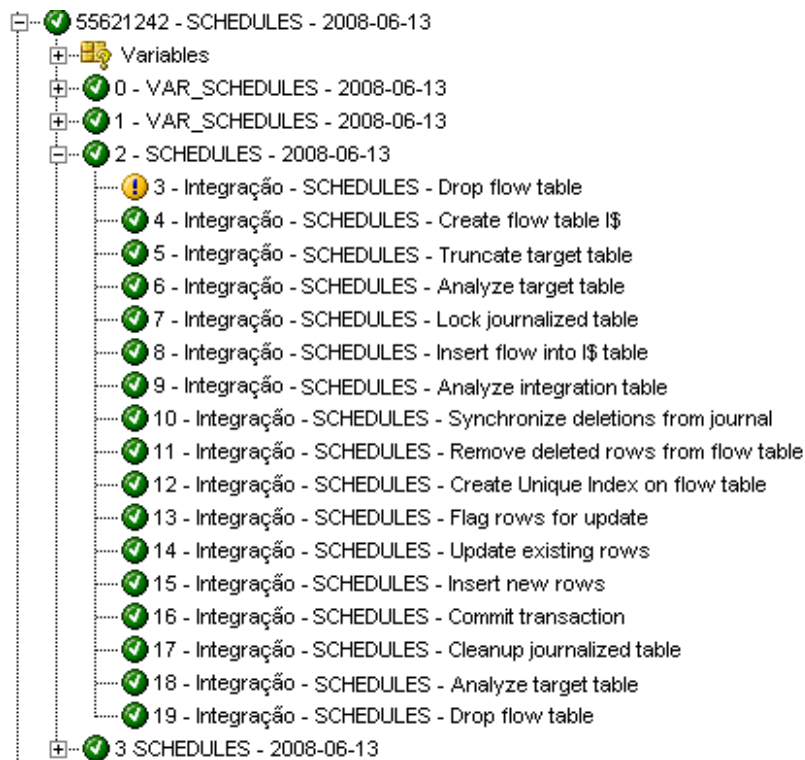


Figura 5.2: Passos a efectuar pela interface de carregamento da tabela de *staging* dos agendamentos.

#### 10. Remover a tabela temporária de integração: processo é concluído.

O quarto, e último, passo corresponde à interface de carregamento da tabela de factos a partir da tabela de *staging*.

Os dados a carregar para a tabela de factos podem ser divididos em três categorias:

- Dados do sistema operacional (tabela de *staging*);
- Dados da tabela dos atributos comuns;
- Dados das dimensões.

O cruzamento das tabelas acima descritas permite o carregamento da tabela de factos. Além desses cruzamentos, é necessário também filtrar a informação proveniente do sistema operacional. Assim, como um agendamento pode não ter episódio associado, pode não existir registo para um determinado agendamento na tabela de atributos comuns. Se for possível, o mapeamento é feito a partir da tabela dos atributos comuns. Se não for possível, então o mapeamento é feito por cruzamento com as dimensões atributos comuns. As dimensões devem ser filtradas de forma a considerar exclusivamente os registos activos (campo do estado do registo). Os mapeamentos das chaves estrangeiras para as dimensões devem ter em atenção a possibilidade do cruzamento poder ser vazio (não

existe um determinado valor na dimensão). Deste modo, o próprio cruzamento controla essa situação, ou então o mapeamento tem o valor -2”(equivalente ao registo do ”não aplicável”em todas as dimensões) de forma a nunca ter valores *nulls* nas tabelas de factos.

O fluxo de dados nesta interface pode ser dividido em dois casos: carregamento de dados entre base de dados e transformação de dados. Estes dois passos correspondem ao ”LT”da arquitectura E-LT do ODI. A extracção dos dados do sistema fonte para a *staging* corresponde ao ”E”. Logo, temos dois KMs para o fluxo de dados da interface de carregamento da tabela de factos:

- O KM de carregamento, neste caso o LKM Oracle to Oracle, que é optimizado para transferência de dados entre bases de dados Oracle;
- O KM de integração, neste caso o IKM FACTS ADW que é usado nas interfaces de carregamento dos factos do ADW.

A arquitectura do ODI está na base deste modelo, pois relativamente a um processo ETL normal o carregamento e a transformação têm uma ordem invertida de execução. Esta arquitectura permite maximizar a eficiência, usando a base de dados de destino e o seu poder de processamento para fazer as transformações necessárias em vez de usar outra base de dados para o efeito.

O IKM FACTS ADW e o LKM Oracle to Oracle têm como principais características:

- Utilizar um dblink (acesso a uma base de dados a partir de outra) para carregar os dados;
- Não remover os dados da tabela de destino;
- Fazer *merge* dos dados (actualizar dados existentes e se não for possível inserir);
- Apagar objectos temporários.

Na Figura 5.3 estão apresentados os diversos passos de execução da interface de carregamento da tabela de factos dos agendamentos. O primeiro passo consiste em criar uma vista sobre os objectos da base de dados de origem a partir de um dblink. Essa vista pretende maximizar a eficiência do carregamento de dados. Criadas as condições na origem para carregar os dados de forma eficiente, é necessário fazer o mesmo no destino. Consequentemente, o próximo passo cria na base de dados de destino uma tabela temporária de integração. A criação da tabela inclui a inserção dos dados da vista criada no primeiro passo.

A seguir, é efectuado um merge para a inserção dos dados na tabela de factos. Um merge actualiza um registo se ele já existir ou insere-o, caso contrário. No fim desse passo é feito *commit* à transacção e é removida a tabela temporária de integração.

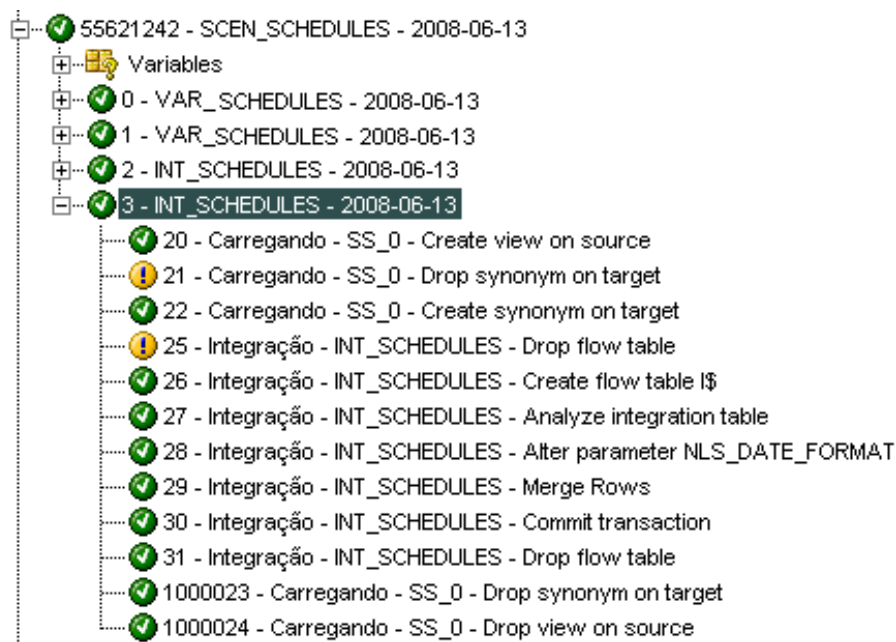


Figura 5.3: Passos a efectuar pela interface de carregamento da tabela de factos dos agendamentos.

### 5.1.2 Testes

Os seguintes testes foram efectuados sobre a estrela dos agendamentos:

- Testes unitários e funcionais;
- Testes de performance;
- Testes de qualidade de dados.

Em termos de testes funcionais, o processo ETL foi executado com diversos carregamentos a partir de uma base de dados usada para testes de desenvolvimento. Essa base de dados tem actualização regular de dados, pelo que é uma excelente plataforma para testes de processos ETL. Foram executados os seguinte casos de teste:

- 5.000 agendamentos que tinham episódios associados;
- 5.000 agendamentos sem episódios associados;
- 80.000 agendamentos escolhidos aleatoriamente;
- Execução do processo durante uma semana com intervalos de execução de 10 minutos com simulação regular de introdução de dados no sistema operacional.

Os resultados dos casos de teste foram todos positivos. O carregamento é efectuado com sucesso e as tabelas de erros não continham registos. No caso do carregamento de 80.000 agendamentos, houve um registo que foi para a tabela de erros porque o registo

tinha um agendamento para o ano 2095. Esse ano não existia na dimensão do tempo pelo que o cruzamento dava erro. Contudo, esse registo contém dados incorrectos já que um agendamento para o ano 2095 não é consistente com a aplicação pelo que esse registo não deveria passar para o *data warehouse*. O processo procedeu de forma correcta ao enviar esse registo para erro.

De forma a garantir que os dados pretendidos eram carregados, foram comparados os identificadores dos agendamentos no *data warehouse* com os extraídos do sistema operacional. Efectivamente, garantimos assim a passagem da informação correcta. De forma a verificar se os factos de um agendamento estavam a ser passados de forma correcta para o *data warehouse* foram testados manualmente alguns valores entre os dois sistemas.

Para verificar a performance do carregamento da estrela, foram efectuados vários carregamentos com um número crescente de registos. Os resultados são apresentados na Figura 5.4 e mostram um crescimento linear da duração relativamente ao número de registos processados. Estes testes foram realizados com uma tabela de factos contendo milhares de registos, de forma a simular um ambiente real, e foram realizados várias vezes de forma a considerar valores médios.

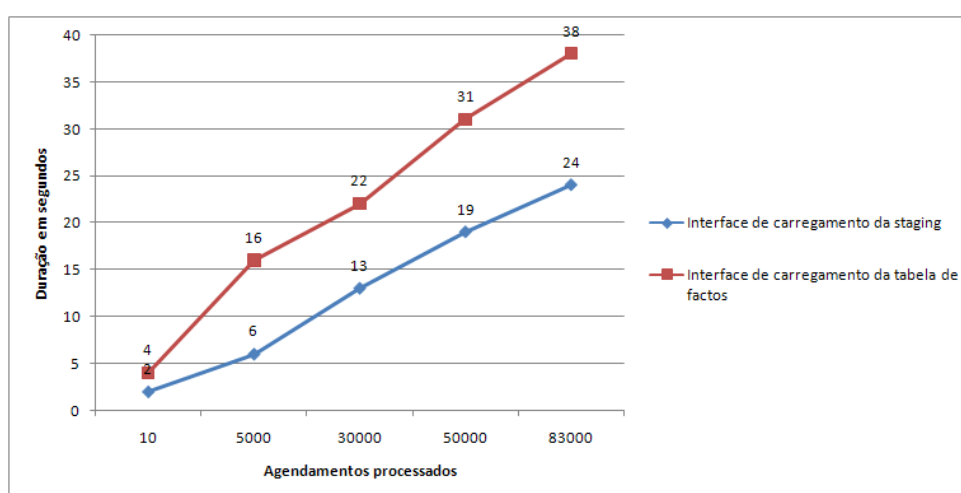


Figura 5.4: Duração do processo ETL dos agendamentos.

Os testes de qualidade de dados foram efectuados recorrendo à criação de invariantes sobre os dados. Uma invariante é um predicado que é sempre verdadeiro durante a execução de uma sequência de operações. Neste caso, as invariantes permitem verificar se os dados provenientes do sistema operacional são coerentes. As principais invariantes criadas para testar o processo foram:

- Agendamentos com data do agendamento superior à data de início;
- Agendamentos requisitados sem data de requisição ou profissional que requisitou;
- Agendamentos sem paciente atribuído;

- Agendamentos cancelados sem data de cancelamento ou profissional que cancelou;
- Agendamentos com data de início de agendamento superior à data prevista de fim.

## 5.2 Estrela das tarefas dos profissionais

### 5.2.1 Desenvolvimento

O processo ETL das tarefas dos profissionais tem vários objectivos:

- Extrair os dados relativos as tarefas de profissionais, no sistema operacional, e carregá-los no *data warehouse*;
- Permitir resolver problemas de performance na duração do processo;
- Retirar carga do sistema operacional;
- Melhorar a performance das interrogações provenientes da interface do utilizador.

Se o primeiro objectivo é um requisito funcional do processo e é consequentemente fundamental, os outros são não-funcionais e representam objectivos desejáveis do processo.

De forma a criar um processo consistente de extracção, transformação e carregamento de dados, o ODI é a escolha mais indicada pois permite-nos a integração total com o resto do processo e grande flexibilidade na criação de fluxos de dados.

Os problemas de carga sobre o sistema operacional estão associados ao facto da informação sobre as tarefas ser extraída exclusivamente do ALERT. No entanto, alguns dos indicadores que se pretendem para a tabela das tarefas já podem ser calculadas com base na informação de algumas estrelas. Todas as tarefas não podem ser, contudo, carregadas a partir do *data warehouse*, deste modo, é possível retirar parte da extracção dos dados do sistema operacional.

Os problemas de performance relativos ao processo ETL podem ser atenuados com o carregamento de dados a partir do *data warehouse* e devem ser reduzidos também a partir de uma maximização da eficiência do próprio processo.

De forma a melhorar a performance das interrogações ao *data warehouse*, é necessário carregar as estruturas agregadas definidas no Capítulo anterior. Estas permitem maximizar a eficiência pesquisas temporais.

Uma das condicionantes relativamente às tarefas dos profissionais está relacionada com o conceito de "cancelado" no sistema operacional. Efectivamente, no ALERT® as tarefas efectuadas pelos profissionais podem ser canceladas. O motivo pode ser um engano, ou a anulação de uma operação. Sendo assim, não faz sentido carregar no *data*

*warehouse* registos que sejam enganos ou cancelamentos de processos. Deste modo, é necessário filtrar os registos cancelados para não serem carregados porque, efectivamente, as tarefas não ocorreram, ou foram canceladas. Mas essa filtragem não é suficiente, pois uma tarefa que podia ser válida no passado e carregada para o *data warehouse*, pode ser cancelada de um instante para o outro e deve ser eliminada do *data warehouse* no próximo carregamento.

O processo ETL que pretende atingir esses objectivos está representado de forma global na Figura 5.5.

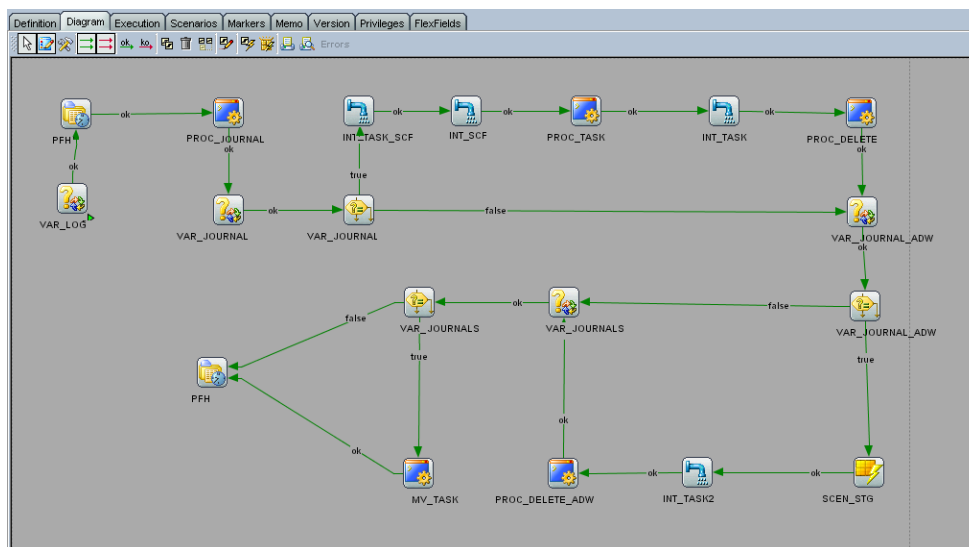


Figura 5.5: Visão geral do processo ETL das tarefas.

O processo ETL das tarefas dos profissionais pode ser dividido em dois sub processos ETL:

- Processo ETL cujos dados são extraídos a partir do sistema operacional ALERT;
- Processo ETL cujos dados são extraídos a partir do ADW.

O processo de carregamento das tarefas dos profissionais a partir do sistema operacional é responsável por carregar, sensivelmente, metade das tarefas. Como nos restantes processos ETL, alguns dos aspectos que são necessários definir são:

- Detecção de actualizações;
- Extração dos dados actualizados;
- Processo de comparação entre *staging* e destino;
- Processo de actualização do destino;

O primeiro processo a definir é o de detecção de actualizações. O principal problema que se coloca, no caso das tarefas dos profissionais, está relacionado com a descentralização da informação. Efectivamente, as tarefas dos profissionais estão dispersas por inúmeras tabelas, no sistema operacional. A solução passa por guardar numa tabela centralizada os dados relativos às tarefas que foram alteradas. A tabela (JOURNAL\_TAREFAS) que guarda esses dados tem a seguinte estrutura:

- Identificador do episódio;
- Nome da tarefa;
- Identificador da tarefa;
- Nome do identificador da tarefa.

Para preencher essa estrutura, é necessário detectar os registos alterados, no sistema operacional. A solução passa por uma tabela de parametrização que guarda informação sobre as tabelas do sistema operacional que correspondem às tarefas. Esta tabela permite auxiliar no carregamento da JOURNAL\_TAREFAS e tem o seguinte formato:

- Nome da vista de "journalização";
- 3 campos com um nome de coluna e 3 campos com um nome de tabela que permitem a partir de uma tabela devolver o identificador do episódio correspondente;
- Nome do identificador da tarefa;
- Filtro SQL sobre a vista de "journalização";
- Nome da tarefa.

Todas as tabelas "journalizadas" relativas às tarefas são guardadas nesta tabela de parametrização. Para cada tarefa que é carregada a partir do ALERT corresponde um registo da tabela de parametrização. Cada registo indica a forma de relacionar a tabala "journalizada" e o identificador do episódio, indica qual o nome da tarefa e o identificador correspondente na tabela. Além disso, como podem existir várias tarefas guardadas na mesma tabela, existe um filtro SQL de forma à filtrar os registos de uma determinada tarefa numa tabela. Por exemplo, um registo possível para esta tabela está representado pela tabela 5.1

Tabela 5.1: Exemplo de registo da tabela de parametrização das tarefas.

|                |     |             |                                      |          |
|----------------|-----|-------------|--------------------------------------|----------|
| ...            | ... | ...         | ...                                  | ...      |
| TABELA_EXEMPLO | ... | ID_TAREFA_1 | WHERE TIPO_TAREFA = 'qualquer coisa' | TAREFA_1 |

De forma a maximizar a eficiência do processamento e a sua manutenção, o processo ETL que carrega os dados do ALERT para a *staging* é feito em código SQL e PL/SQL. Em termos de fluxo de dados, este pode ser representado pela Figura 5.6.

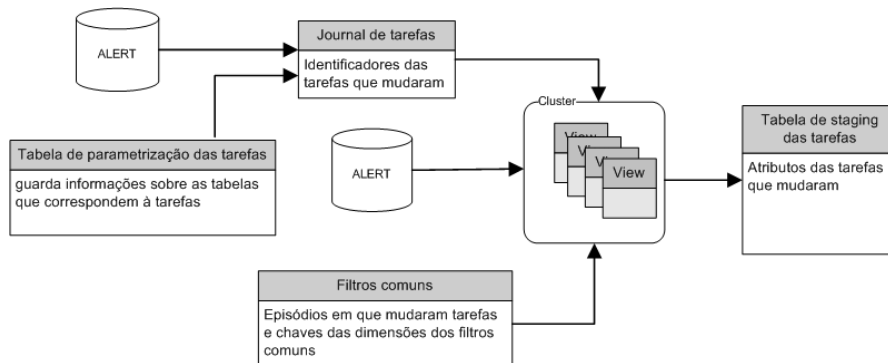


Figura 5.6: Extração e carregamento de tarefas para a tabela de *staging* a partir do ALERT®.

A extração dos dados é feita a partir de vistas criados sobre o ALERT, cruzadas com os identificadores das tarefas que foram alteradas, dos episódios e dos seus common filters associados. Essas vistas servem de base para o carregamento para a tabela de *staging* da tabela de factos. De forma a ajudar no carregamento da tabela de *staging* a partir das vistas, foram criadas duas estruturas:

- **Tabela da parametrização das vistas:** identifica as vistas a usar no processo de carregamento, assim como as condições SQL sobre elas;
- **Tabela de parametrização dos campos da *staging*:** para cada campo da tabela de *staging* e para cada tipo de tarefa, indica qual a vista e campo da vista que mapeia esse campo.

O processo de comparação e actualização dos dados é feito recorrendo ao motor do ODI e a código PL/SQL. Existe uma interface de carregamento da tabela de factos a partir da tabela de *staging*. Esta interface carrega unicamente as tarefas não canceladas que se encontram na tabela de *staging*. Ainda fica por resolver o problema dos cancelados, já que temos de eliminar possíveis tarefas que tenham sido canceladas desde o último processamento e que se encontrem actualmente na tabela de factos. Na tabela de *staging* existem todas as tarefas alteradas desde o último carregamento, inclusive as canceladas. Logo, é possível comparar as tarefas canceladas na *staging* com a table de factos. Esta comparação é feita por um procedimento em PL/SQL que apaga as tarefas canceladas que estejam na tabela de *staging*. Outro aspecto importante no cancelamento de tarefas é o facto de possuírem uma relação pai-filho entre elas, isto é, algumas tarefas são dependentes de outras. Se for cancelada a requisição de uma análise por exemplo,

o resultado dessa análise fica automaticamente cancelado. Esse cancelamento pode não existir de forma explícita e persistente, no sistema operacional, pelo que para garantir a consistência dos dados, essa operação é feita no processo ETL. Assim o procedimento que apaga os cancelados também apaga todas as tarefas que são dependentes de tarefas canceladas. A relação de dependência entre tarefas está definida na dimensão do tipo de tarefa definida no Capítulo anterior.

Os fluxos de dados da interface de carregamento da tabela de factos são definidos pelos seguintes KMs:

- LKM Oracle to Oracle;
- IKM FACTS ADW.

Estes knowledge modules têm o mesmo comportamento da interface de carregamento da tabela de factos dos agendamentos e estão definidos na secção 5.1.1.

Assim, o processo para carregar a tabela de factos, a partir da base de dados do ALERT, segue os seguintes passos:

1. Procedimento PL/SQL que identifica as tarefas modificadas e carrega uma estrutura com os identificadores dessas tarefas (JOURNAL\_TAREFAS);
2. Actualização da variável que conta o número de registos da estrutura do passo anterior;
3. Avaliação da variável: tem de ser superior à zero;
4. Interface de carregamento da tabela de *staging* (do ALERT) dos atributos comuns relativos às tarefas dos profissionais;
5. Interface de carregamento da tabela de *staging* (do ADW) dos atributos comuns relativos às tarefas dos profissionais;
6. Procedimento PL/SQL que carrega a tabela de *staging* a partir de vistas sobre o sistema operacional;
7. Interface de carregamento da tabela de factos, a partir da tabela de *staging*;
8. Procedimento que apaga as tarefas canceladas;

Além do processo ETL que extrai os dados do sistema operacional, existe o processo que carrega as restantes tarefas a partir dos dados existentes nas estrelas do ADW. Para detectar as actualizações nas tabelas de factos, estas têm de ser "journalizadas". Um novo processo de "journalização" tem de ser definido pois é necessário incluir tabelas de um novo modelo de dados e de uma nova base de dados. Este processo de

”journalização” envolve a criação de um novo *subscriber* que detecta as alterações efectuadas nas tabelas de factos que contêm registos com tarefas de profissionais. Por exemplo, a tabela de factos das análises e dos exames têm de ser ”journalizadas”. pois uma requisição de análise ou exame corresponde a uma tarefa.

Na Figura 5.7 é representado o fluxo de dados geral do carregamento de tarefas para a tabela de factos a partir do ADW.

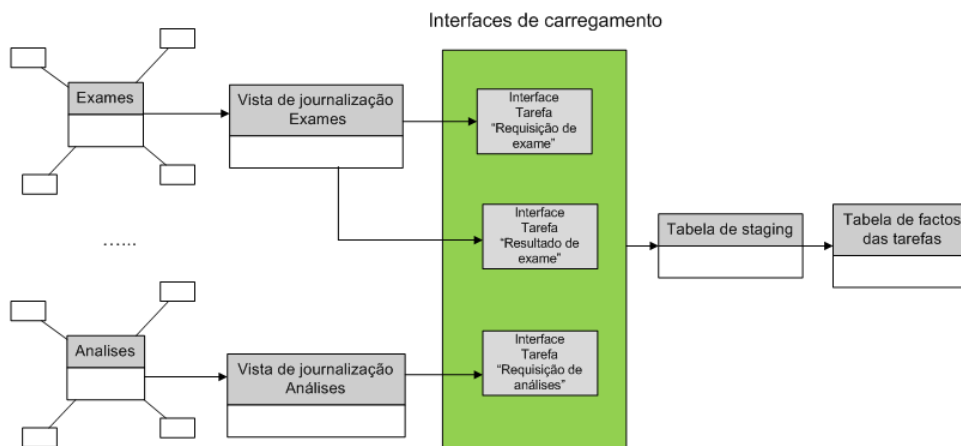


Figura 5.7: Carregamento de tarefas para a tabela de factos a partir do ADW.

As vistas de ”journalização” sobre essas tabelas permitem, a cada processamento, visualizar os dados alterados nas tabelas de factos desde o último processamento. De forma a processar e carregar os registos alterados, interfaces do ODI permitem definir o carregamento numa tabela de *staging* de factos das tarefas. Cada tarefa necessita de ser carregada por uma interface diferente pois a cada tarefa está associado um mapeamento diferente. Nas interfaces estão definidos os cruzamentos com as dimensões e os filtros sobre as tabelas fonte. É, efectivamente, necessário filtrar todas as dimensões com as quais se cruza a vista de ”journalização”, de forma a permitir exclusivamente a passagem de registo com estado SCD activo.

No ODI é possível alterar os KMs, de forma a personalizar o fluxo de dados. Para a implementação do fluxo de dados dessas interfaces foi necessário criar um novo IKM, pois nenhum existente preenchia os requisitos. Efectivamente, as interfaces em causa são bastante simples e carregam uma tabela de *staging* em que são sempre inseridos novos dados. Consequentemente, o processo de actualização nunca é feito sobre essa tabela, pelo que o IKM não precisa desse passo. De forma a otimizar a eficiência do processo ETL, alterou-se o processo ETL de forma a que nunca faça *update* e faça sempre *insert*. Este novo IKM, o IKM TAREFAS ADW tem as seguintes características:

- não verifica a existência do registo na tabela de destino;

- efectua sempre um *insert*;
- não faz *truncate* à tabela de destino (vários interfaces necessitam de carregar a tabela).
- inclusão do *hint ordered* para a geração do código.

De forma a garantir a inserção de dados novos a cada execução do processo ETL, é necessário fazer um truncate à tabela de *staging* antes de executar o primeiro interface.

A inclusão de um *hint* deve-se a um problema encontrado na implementação das interfaces relativamente ao plano de execução do SQL gerado. Um plano de execução é uma sequência de passos, usado para aceder ou modificar a informação numa base de dados relacional implementada em SQL. Visto que o SQL é declarativo, existem inúmeras alternativas para executar uma determinada interrogação, com níveis de performance diferentes. Quando uma interrogação é submetida à base de dados, o otimizador avalia os possíveis planos e retorna aquele que ele considera a melhor alternativa. No entanto, os optimizadores são imperfeitos e os planos podem não ser os mais adequados pelo que existe um mecanismo que permite obrigar o optimizador a escolher um determinado tipo de plano. Este mecanismo é a inclusão de um *hint* na interrogação. O problema identificado era relativo à execução de um *hash join*, operação com enorme custo, em vez da execução de um *nested loop*. O *hint ordered* permite resolver essa situação e obrigar o optimizador a escolher um *nested loop* em vez de um *hash join*. Foi adicionado ao próprio IKM este *hint*, de forma a que todas as interfaces o usem nas interrogações produzidas.

Os passos de execução da interface de carregamento da *staging* são representados na Figura 5.8. Como já foi referido, o IKM foi reduzido ao mínimo de modo a maximizar a performance. Assim sendo, o primeiro passo consiste em remover a tabela de integração, caso ela exista. De seguida, é criada a tabela de integração, a partir dos cruzamentos definidos na interface, e são inseridos os dados existentes. Não existe nenhum controlo de *update* pelo que o próximo passo é a inserção directa dos dados na tabela de destino. A transacção é tornada persistente por um commit e, finalmente, a tabela de integração é removida.

Após a inserção dos dados na tabela de *staging*, procede-se à inserção dos dados na tabela de factos, a partir de uma interface definida no ODI. A interface só contém um cruzamento com a dimensão do tipo de tarefa. Os restantes cruzamentos não necessitam de ser efectuados, pois já foram feitos aquando do carregamento das estrelas no ADW. A interface inclui um filtro sobre a tabela de *staging* que impede o carregamento das tarefas canceladas. Este filtro deve-se ao facto de considerarmos que tarefas canceladas não correspondem a uma tarefa real na aplicação, pelo que não devem ser contabilizadas. Nas estrelas, os registos cancelados são incluídos devido à granularidade e dos requisitos

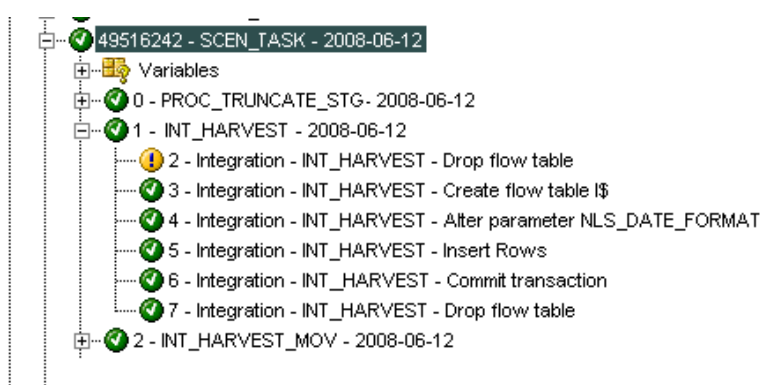


Figura 5.8: Passos a efectuar pela interface de carregamento da tabela de *staging* das tarefas.

de negócio serem diferentes. O IKM usado para esta interface é o IKM FACTS ADW cuja descrição já foi feita na secção 5.1.1.

É ainda mais evidente, nesta altura, os benefícios de carregar os dados a partir do ADW em vez de carregar a partir do ALERT. As interfaces de carregamento são muito simples, devido ao facto dos cruzamentos necessários com as dimensões já terem sido efectuados. Além disso, as operações situam-se todas na mesma base de dados, o que permite maximizar ainda mais a eficiência.

Da mesma forma que era necessário remover os registos cancelados no carregamento a partir do sistema operacional, esta operação verifica-se também aqui. Os pressupostos são os mesmos, podendo existir tarefas na tabela de factos que foram canceladas entre dois processos ETL e que necessitam de ser removidas. O passo a seguir ao carregamento da tabela de factos é um procedimento PL/SQL que remove essas tarefas assim como as tarefas que dependem delas.

De forma resumida, o processo ETL para as tarefas carregadas a partir do ADW segue os seguintes passos:

1. Estender a janela de processamento e bloquear o *subscriber*, de forma a processar os dados que foram alterados desde o último processamento;
2. Actualizar o valor da variável que conta o número de registos das vistas de "journalização";
3. Avaliar o valor da variável: superior à zero;
4. Fazer *truncate* à tabela de *staging*;
5. Executar interfaces de carregamento da tabela de *staging* para cada tipo de tarefa;
6. Executar interface de carregamento da tabela de factos;
7. Executar procedimento PL/SQL que remove as tarefas canceladas;

8. Desbloquear o *subscriber* e apagar os dados dos *journals*;

Um problema que surgiu durante a implementação do processo está relacionado com o processo de carregamento das tabelas agregadas sobre a tabela de factos. Nomeadamente, questões de performance surgiram na implementação. Desta forma, foi necessário avaliar a melhor forma de garantir o acesso à informação das agregadas mantendo um elevado nível de performance. Resolveu-se dividir o processo em duas fases:

- Carregamento da tabela agregada à hora durante o processo ETL global;
- Carregamento das tabelas agregadas ao dia e hora num processo separado a executar uma vez por dia.

Esta decisão tem como base o facto da tabela agregada à hora ser de importância fundamental para os utilizadores de ADW. Além da informação ser fundamental, o seu acesso em tempo real é uma mais valia muito apreciada. De forma a manter o mesmo nível de funcionalidade relativamente à antiga estrela, a informação da agregada à hora deve ser carregada na mesma altura que a tabela de factos.

Por outro lado a informação agregada ao dia e mês não existia na antiga estrela, pelo que a sua inclusão no processo global não faz sentido devido ao peso das operações de carregamento.

A inclusão da agregada da hora no processo pode ser feita através de duas opções:

- Remover os dados da tabela e inserir os dados agregados à cada processo ou,
- Implementar um mecanismo de actualização da tabela agregada a partir da tabela de factos.

A primeira opção seria de um custo enorme em termos de performance pois estamos a falar de uma tabela de factos com milhões de registos. A segunda opção implicaria a implementação de um mecanismo de grande complexidade e propício a erros. No entanto, existe uma alternativa para implementar a segunda opção que não exige uma implementação complexa: uma *materialized view*.

As *materialized views* são uma extensão das vistas tradicionais. Efectivamente, as *materialized views* permitem visualizar os dados de uma ou mais tabelas fontes. Contudo, têm uma abordagem diferente já que o resultado da interrogação à vista é guardado, de forma persistente, na base de dados, e pode ser actualizado de forma periódica. As vantagens de uma *materialized view* são relacionadas essencialmente com questões de performance:

- Menos leituras físicas;

- Menos operações de escrita;
- Menos consumo de CPU;
- Melhor tempo de resposta.

As *materialized views* são, conseqüentemente, otimizadas para operações intensivas de leitura, que é exactamente aquilo que procuramos. Desta forma ao criar uma *materialized view* que representa a tabela de agregação à hora das tarefas, podemos beneficiar das vantagens da agregada em tempo real sem um elevado custo de processamento.

Algumas das especificidades da *materialized view* das tarefas agregada à hora são:

- Criação de uma log (*snapshot*) sobre a tabela de factos de forma a poder actualizar a *materialized view*;
- Actualização da view de forma manual evitando assim problemas de concorrência.

O último passo do processo ETL consiste na execução do procedimento PL/SQL que actualiza a *materialized view*. Este procedimento tenta fazer um *fast refresh* e se não conseguir faz um *complete refresh*. O *fast refresh* tenta usar a log que contém as alterações feitas para actualizar a *materialized view*. Um *complete refresh* descarta o uso de logs e faz o mesmo que faria numa vista normal (uma interrogação total à tabela fonte).

As outras tabelas de agregação são carregadas uma vez por dia, de forma a minimizar o impacto em termos de performance. Foi considerado o uso de *materialized views* para as agregadas ao dia e mês mas, não sendo essenciais em real time, privilegiou-se a performance. O processo de carregamento dessas tabelas está incluído num *package* ODI com duas interfaces, representado na Figura 5.9.

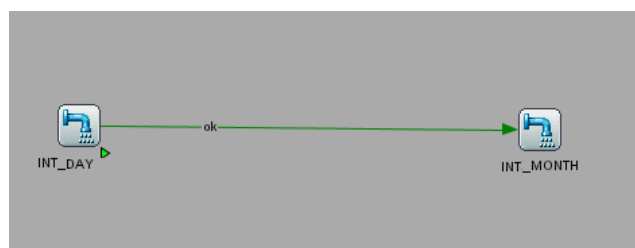


Figura 5.9: Visão geral do Processo ETL das tarefas agregadas.

As interfaces de carregamento dessas duas tabelas é efectuada usando o IKM Oracle Incremental Update alterado:

- Só é feito *insert* aos dados;
- A tabela de destino é removida antes de começar o carregamento.

### 5.2.2 Testes

O processo de testes sobre a estrela das tarefas segue o mesmo modelo dos agendamentos, pelo que foram efectuados testes unitários e funcionais, testes de qualidade de dados e testes de performance.

Os testes unitários realizados, no caso das tarefas, incidem sobre a avaliação de cada etapa do processo ETL, em termos de comparação entre input e output:

- Caso de teste em que são alterados vários registos no sistema operacional: verificação da etapa de detecção de actualizações, comparando os registos actualizados com os registos na tabela JOURNAL\_TAREFAS;
- Caso de teste em que comparamos os registos da tabela JOURNAL\_TAREFAS com os registos da tabela de *staging* das tarefas;
- Caso de teste em que comparamos os registos da tabela de *staging* das tarefas com os registos da tabela de factos das tarefas;
- Caso de teste em que são alterados registos das estrelas do ADW: verificação da etapa de detecção de actualizações, comparando os registos actualizados com os registos das vistas de "journalização" sobre as tabelas de factos do ADW;
- Caso de teste em que comparamos os registos das vistas de "journalização" e da tabela de *staging* do ADW;
- Caso de teste em que comparamos os registos da tabela de *staging* do ADW e da tabela de factos;
- Caso de teste em que comparamos os registos da tabela de factos com os das tabelas agregadas.

Estes casos de teste foram realizados para diversos carregamentos sequenciais. De forma a verificar que os inputs e outputs eram iguais, foram comparadas as chaves primárias. Os resultados dos testes foram positivos, garantindo desta forma a funcionalidade principal do processo extracção e carregamento de dados. Para os testes funcionais foram usados ainda alguns casos de teste em que se criou uma tarefa, no ALERT, e verificou-se a sua passagem correcta para o ADW, após o processo ETL. De forma paralela aos testes dos agendamentos, deixou-se em execução o processo ETL durante uma semana com intervalos de 10 minutos e introdução regular de dados.

Durante a fase de testes, foi dado grande ênfase aos testes de performance. Os testes de performance foram realizados sobre as principais etapas do processo ETL, de forma a analisar detalhadamente o comportamento do processo.

## Processo ETL do ADW PRIVATE PRACTICE

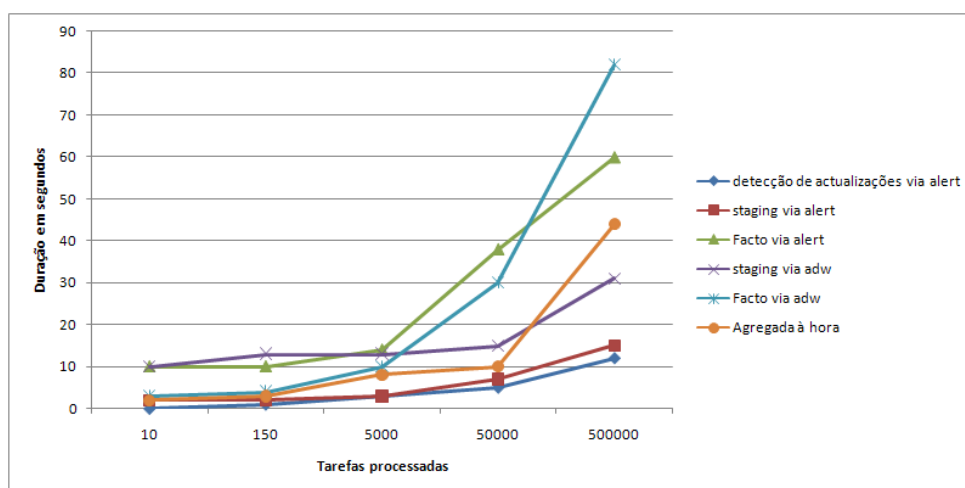


Figura 5.10: Duração do processo ETL das tarefas dos profissionais.

Os testes foram efectuados com uma estrela carregada com mais de um milhão de tarefas. Comparativamente ao processo que estava implementado anteriormente, verifica-se uma franca melhoria na duração (na ordem dos segundos). Além disso, podemos ver que o processo é bastante rápido com tempos de execução semelhantes até ao processamento de 5000 tarefas. Mas o processo ETL corre em real time, de 10 em 10 minutos, pelo que o número de tarefas executados durante esse período dificilmente chegará a valores superiores à ordem dos milhares. Consequentemente, os testes de performance tiveram resultados francamente positivos.

De forma a garantir a qualidade dos dados extraídos e carregados para o *data warehouse*, recorreu-se sobretudo a invariantes. As principais invariantes usadas, no caso da estrela das tarefas, são:

- Tarefa efectuada sem profissional associado;
- Tarefa efectuada sem paciente associado;
- Tarefa efectuado com um tipo de tarefa indefinido ou não aplicável;
- Tarefa efectuado com data superior à data actual.

Outros tipos de invariantes poderiam ter sido especificados, nomeadamente sobre o *workflow* das tarefas. Por exemplo, uma invariante que verifica que se a data de execução de uma requisição é inferior à data do resultado associado. Devido à complexidade das invariantes envolvidas, ao número de tarefas envolvidas (aproximadamente 80) e ao prazo de desenvolvimento, não foi possível especificar e implementar invariantes deste tipo.

A monitorização das tabelas de erros durante o processo ETL também foi efectuada para garantir a qualidade e consistência dos dados. Os testes de qualidade de dados foram

de forma global positivos, detectando alguns dados incorrectos ou incompletos. Efectivamente, os testes foram efectuados sobre uma base de dados do ALERT que está dedicada a testes de desenvolvimento e que portanto tem muitos erros de dados. Foi aliás uma óptima plataforma para detectar se o controlo de qualidade de dados estava funcional.

## **5.3 Adaptações do modelo actual**

### **5.3.1 Desenvolvimento**

Um dos primeiros pontos a resolver é o da internacionalização do produto. É necessário adaptar o produto a diferentes mercados e um dos problemas mais importantes, no caso do ADW, é a consistência da informação das suas principais dimensões. Efectivamente, só as dimensões guardam informação textual que pode mudar de língua para língua.

O ALERT® tem um sistema de controlo de línguas sendo que todos os conteúdos que existem são traduzidos para diversas línguas, ao nível da base de dados. Deste modo, a maior parte das dimensões terá a informação traduzida já que actualmente o ADW é capaz de carregar conteúdos com diferentes línguas.

O problema reside nas dimensões que não carregam o seu conteúdo directamente do sistema operacional. Esses casos incluem as dimensões da geografia, da demografia e do tempo. A última é fácil de controlar, já que o próprio motor de base de dados da Oracle permite gerir descrições sobre datas em diversas línguas. Já existe um script bastante flexível que permite gerar a dimensão do tempo no ADW. No caso da demografia também é fácil de resolver já que a informação contida na dimensão não é muito grande. Só é necessário efectuar algumas traduções.

O caso da geografia é mais complexo, já que a informação deve incluir todos os códigos postais possíveis, assim como a informação associada a esse código postal. No caso de Portugal, essa informação é proveniente de ficheiros CSV (comma separated value) oficiais e públicos dos CTT. Assim, para carregar a dimensão da geografia com dados americanos é portanto necessário uma fonte oficial de informação.

A solução encontrada foi a utilização de um software comercial que disponibiliza esse tipo de informação em vários formatos, sendo o CSV um deles. Foi necessário recorrer a um software comercial, já que ao contrário do que acontece aqui em Portugal, os Estados Unidos, nomeadamente os correios, não disponibilizam de forma pública esse tipo de informação. Como precisamos de informação consistente, decidiu-se recorrer a um produto comercial. Embora existissem ficheiros na web com informação geográfica, estes não nos davam garantias quanto a sua exactidão e fiabilidade.

Parte da estrutura do ficheiro CSV é apresentada na Figura 5.11. Como esse ficheiro não contém informação sobre o nome do estado, é necessário carregar os dados dos nomes dos estados através de outro ficheiro, que contém o nome e a abreviatura do estado. O

| #         | Field Name | Data Type   | Description                                               |
|-----------|------------|-------------|-----------------------------------------------------------|
| 01        | ZipCode    | Char(5)     | 00000-99999 Five digit numeric ZIP Code of the area.      |
| 02        | City       | VarChar(35) | Name of the city as designated by the USPS.               |
| 03        | State      | Char(2)     | 2 letter state name abbreviation.                         |
| 04        | CountyName | VarChar(45) | Name of County or Parish this ZIP Code resides in.        |
| 05        | AreaCode   | VarChar(55) | The telephone area codes available in this ZIP Code.      |
| • • • • • |            |             |                                                           |
| 13        | CountyFIPS | Char(3)     | FIPS code for the County/Parish this ZIP Code resides in. |
| • • • • • |            |             |                                                           |
| 18        | StateFIPS  | Char(2)     | FIPS code for the State this ZIP Code resides in.         |

Figura 5.11: Formato do ficheiro CSV da geografia americana.

carregamento usa tabelas auxiliares para carregar os diversos elementos do ficheiro, para depois relacionar a informação na dimensão. O script de carregamento é composto pelos seguintes passos:

1. Criar uma tabela auxiliar de zipcodes e carregar todos os dados contidos no ficheiro CSV que contém os zipcodes;
2. Criar uma tabela auxiliar de estados e carregar o nome e a abreviatura de todos os estados, a partir de outro ficheiro CSV;
3. Criar uma tabela auxiliar que cruza a informação das duas tabelas anteriores;
4. Apagar a sequência da dimensão da geografia e criar uma nova sequência;
5. Remover os dados da dimensão;
6. Inserir os dados na dimensão da geografia, a partir da tabela auxiliar;
7. Inserir o valor "não aplicável" na dimensão.

Além destes passos e já que o ficheiro não contém identificadores para as cidades da geografia americana, é necessário criar esse identificador doutra forma. Logo, após o carregamento da dimensão da geografia, é necessário executar um script de actualização que ordena os dados por cidade de forma a atribuir à cada um identificador único sequencial.

Os novos atributos identificados no Capítulo anterior inserem-se nos processos ETL actuais já existentes. Estes atributos incluem factos para a tabela de factos dos episódios e das tarefas: data de co-sign, preço da consulta, pedido de co-sign, entre outros. Desta forma no processo ETL dos episódios e das tarefas, os campos foram acrescentados nos mapeamentos das interfaces de carregamento das tabelas de *staging* e das interfaces de carregamento das tabelas de factos.

No caso das datas foi necessário ter em conta o formato a guardar na base de dados. A base de dados do ALERT guarda actualmente as datas no formato *timestamp with timezone*, que guarda informação sobre a data do registo mas também sobre a *timezone* associada. Desta forma, a apresentação da data tem de sofrer uma conversão segundo a *timezone* em que queremos apresentar a informação. Actualmente, essa conversão é feita através de um mapeamento apropriado no processo ETL: para o mercado português, por exemplo, é efectuado um *cast* da data para a *timezone* de Lisboa.

Um dos novos atributos criados representa o estado do pagamento de uma consulta. No ADW, os estados dos processos são guardados numa dimensão de estados. Assim, é necessário para carregar esses estados na tabela de factos, criar os estados correspondentes na dimensão. Os estados a adicionar de forma estática na dimensão são:

- Pagamento da consulta efectuada;
- Pagamento da consulta por efectuar;
- Não aplicável.

Estes estados correspondem ao estado do pagamento na estrela dos episódios. O cruzamento é feito na interface de carregamento da tabela de factos dos episódios.

### 5.3.2 Testes

Os testes sobre estes novos desenvolvimentos foram essencialmente testes funcionais e unitários.

No caso do carregamento da geografia, foram considerados vários casos de teste, de forma a testar funcionalmente o carregamento. Foram escolhidos, aleatoriamente, algumas localizações geográficas presentes nos ficheiros de entrada, e essas localizações foram comparadas com os dados da dimensão. A comparação incidiu na correspondência dos campos da cidade, *county*, estado. O resultado foi positivo com um carregamento correcto dos dados presentes nos ficheiros.

Para os atributos adicionados às tabelas de factos, foram efectuados testes funcionais, de forma a verificar se os campos eram carregados e preenchidos de forma correcta no *data warehouse*, e testes unitários que verificavam em cada etapa do processo, o input e o output produzido.

## 5.4 Resumo ou Conclusões

A implementação do processo ETL para o ADW PRIVATE PRACTICE inclui a criação de vários processos, a integrar no processo geral ETL existente, e da especificação e execução dos respectivos testes .

O primeiro processo envolveu um mecanismo de carregamento dos dados para a estrela dos agendamentos e foi definido através do Oracle Data Integrator. A sequência de execução do processo inclui a detecção de actualizações no ALERT, extracção de dados para uma tabela de *staging*, e carregamento para a tabela de factos dos agendamentos no *data warehouse*. Esse carregamento inclui as transformações a efectuar aos dados.

O processo ETL das tarefas envolveu a separação da extracção dos dados em duas fases: extracção a partir do ALERT e a partir do ADW. Processos de detecção de actualização foram definidos para os dois sistemas fontes usando a funcionalidade de Change Data Capture do Oracle Data Integrator. O carregamento das tabelas de *staging* respectivas foi efectuado por meio de vistas e código PL/SQL, no caso dos dados provenientes do ALERT, e por Oracle Data Integrator, para dados provenientes do ADW. O carregamento da tabela de agregação à hora foi otimizado recorrendo a uma *materialized view*. O carregamento das agregadas ao dia e mês foram definidas num processo separado à executar uma vez por dia, de forma a reduzir a carga sobre o sistema.

Foi necessário ainda criar um script SQL de carregamento dos dados da dimensão da geografia americana a partir de um ficheiro CSV e alterar os processos ETL para as estrelas existentes de forma a incluir o carregamento dos novos factos.

Foram efectuados ainda diversos testes de forma a validar a implementação efectuada: testes funcionais, unitários, de performance e qualidade de dados. Os testes efectuados foram positivos tanto em termos de funcionalidade como de performance. No caso da implementação da estrela das tarefas, a performance atingida superou os objectivos iniciais. Efectivamente, a estrela carrega mais informação e a um nível de granularidade mais baixo e com um desempenho melhor que o antigo processo.

## Capítulo 6

# Conclusões e trabalho futuro

### 6.1 Satisfação dos objectivos

Relativamente ao trabalho realizado, todos os objectivos iniciais foram atingidos. Foi ainda conseguido realizar trabalho para além do que estava planeado.

Uma análise aprofundada do produto ALERT® PRIVATE PRACTICE, do ADW e das suas limitações, assim como uma análise dos requisitos de negócio no contexto de clínicas privadas e consultórios, permitiu definir os indicadores possíveis de extrair, armazenar e apresentar no *data warehouse*. Os indicadores identificados são principalmente indicadores clínicos devido ao sistema fonte ser um produto dedicado à execução de processos clínicos numa instituição de saúde.

No *data warehouse*, foi criado um novo modelo de dados, de forma a contemplar esses novos indicadores. No desenvolvimento do ADW, o modelo de dados é comum a todos os produtos. Por essa razão, o modelo de dados foi alterado de forma a ser genérico, no sentido em que é transversal a qualquer tipo de produto ADW, e a ser específico no sentido em que contém estruturas que armazenam dados exclusivos ao produto ALERT® PRIVATE PRACTICE. O modelo de dados já inclui indicadores sobre análises, exames, diagnósticos, entre outros. A esta informação, foi necessário acrescentar novos indicadores sobre agendamentos de consultas, criar uma estrutura nova e indicadores específicos que integram-se nas estruturas já existentes. De forma a responder a novos requisitos de negócio e a problemas de desempenho, foi necessário reformular de raiz a estrutura que armazena as tarefas dos profissionais. Esta é a estrutura mais complexa e de maior dimensão do ADW.

De forma a armazenar os dados no *data warehouse*, foi necessário implementar um processo ETL, sobre os novos desenvolvimentos do modelo de dados, que se integra com o processo geral já existente. A implementação do processo ETL começa pelo desenvolvimento dum processo de detecção de actualizações no sistema fonte e de definição das fontes de dados a extrair e dos seus cruzamentos e filtros. Além disso, envolve a comparação entre a fonte e o destino, o mapeamentos de dados para o destino, assim como o controlo de erros de carregamento e qualidade de dados. Adaptações foram efectuadas, para garantir o carregamento e a consistência do *data warehouse*, na língua inglesa. Nomeadamente, o carregamento da informação da geografia americana, que foi implementada recorrendo a informação que não faz parte do sistema fonte. O processo ETL das tarefas dos profissionais foi implementado recorrendo a duas fontes de dados, o próprio *data warehouse* do ADW e a base de dados do ALERT® PRIVATE PRACTICE.

A eficiência do processo ETL foi testada, recorrendo a diversos casos de teste que carregavam um número crescente de registos. Foi avaliada, sobretudo, a duração das diversas etapas de processamento. Neste ponto, foram superados os objectivos iniciais. Além de garantir a eficiência dos processos ETL desenvolvidos, foi melhorada a performance do processo ETL das tarefas dos profissionais relativamente ao processo existente.

A garantia de qualidade de dados do *data warehouse* foi verificada através de diversos mecanismos. Umas invariantes foram criadas sobre o conteúdo dos dados do sistema fonte. Estas permitiram verificar a consistência dos dados processados. Foram criados diversos casos de testes unitários para garantir a qualidade e integridade dos dados ao longo de todo o processo. Os testes unitários gerados envolvem a comparação entre os dados de entrada e os dados de saída duma etapa do processo. Outro método de controlo de dados implementado envolve a utilização de tabelas de erros. As tabelas de erros guardam registos que não satisfazem as regras de integridade e que provocam erros durante um carregamento de dados do processo ETL. Desta forma, garante-se a integridade dos dados no *data warehouse*.

Em suma, o trabalho realizado satisfaz os objectivos pretendidos. Aliás o trabalho realizado está incluído na próxima versão dos produtos ALERT® e ADW, a serem instaladas brevemente.

## 6.2 Trabalho futuro

O trabalho desenvolvido sobre o ADW PRIVATE PRACTICE, apesar de ter atingido os objectivos, não está concluído. Efectivamente, estão a ser desenvolvidas as novas funcionalidades na interface do utilizador, de forma a permitir a consulta de informação.

Relativamente aos desenvolvimentos ao nível da base de dados e de processo ETL, existem também diversas perspectivas futuras e que incidem directamente sobre o trabalho realizado neste projecto.

Uma das limitações do trabalho realizado neste projecto está relacionado com a integração de sistemas fontes que não sejam a base de dados do ALERT®. A partilha de dados entre sistemas de informação é um processo complexo que envolve decisões superiores. Desta forma, alguns dos indicadores administrativos e financeiros não puderam ser carregados, no *data warehouse*, devido a essa limitação. Uma perspectiva futura envolve a integração de outros sistemas de informação no *data warehouse*. Uma das soluções mais viáveis envolve a integração da informação de sistemas CRM e ERP que estão a ser desenvolvidos, internamente, dentro da gama da família de produtos ALERT®.

Outra perspectiva de trabalho futuro envolve a manutenção do processo ETL e do modelo de dados, consoante a evolução do(s) sistema(s) fonte(s). Novas versões do ALERT® PRIVATE PRACTICE serão desenvolvidas, pelo que o modelo de dados e as próprias funcionalidades do sistema fonte estão em constante mudança. Logo, existe uma necessidade contínua de novos desenvolvimentos, de forma a manter o *data warehouse* sempre compatível com o sistema fonte. Além da manutenção do ADW PRIVATE PRACTICE, existem perspectivas de melhoramento do produto. Sendo o trabalho realizado a primeira versão dum produto, poderá ter muito feedback por parte dos clientes, em termos de melhoramentos ou novas funcionalidades.

## Conclusões e trabalho futuro

# Referências

- [Ada06] Christopher Adamson. *Mastering Data Warehouse Aggregates: Maximizing Star Schema Performance*. Wiley, John and Sons, Incorporated, First edition, 2006.
- [AlSC08] S.A. ALERT life Sciences Computing. Documento executivo da alert, Maio 2008. <http://www.alert-online.com/>.
- [CA97] M.J. Corey and M. Abbey. *Oracle data warehousing: A Practical guide to successful data warehouse analysis, build, and roll-out*. 1997.
- [Dat99] C. J. Date. *An introduction to database systems (7th ed.)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.
- [DDD<sup>+</sup>04] B. Dageville, D. Das, K. Dias, K. Yagoub, M. Zait, and M. Ziauddin. Automatic SQL tuning in oracle 10g. *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pages 1098–1109, 2004.
- [GMWU99] H. Garcia-Molina, J. Widom, and J.D. Ullman. *Database System Implementation*. Prentice-Hall, Inc. Upper Saddle River, NJ, USA, 1999.
- [Inm96] W. H. Inmon. *Building the data warehouse (2nd ed.)*. John Wiley & Sons, Inc., New York, NY, USA, 1996.
- [Kar] R. Karel. The Forrester Wave: Enterprise ETL, Q2 2007. *Forrester Research*.
- [KC04] Ralph Kimball and Joe Caserta. *Data Warehouse ETL Toolkit: Practical Techniques for Extracting, Cleaning, Conforming, and Delivering Data*. Wiley, John and Sons, Incorporated, First edition, 2004.
- [Kim97] Ralph Kimball. A dimensional modeling manifesto. *DBMS*, 10(9):58–70, 1997.
- [KR02] Ralph Kimball and Margy Ross. *The Data Warehouse Toolkit*. Wiley, John and Sons, Incorporated, Second edition, 2002.
- [KRT<sup>+</sup>98] Ralph Kimball, Laura Reeves, Warren Thornthwaite, Margy Ross, and Warren Thornwaite. *The Data Warehouse Lifecycle Toolkit: Expert Methods for Designing, Developing and Deploying Data Warehouses with CD Rom*. John Wiley & Sons, Inc., New York, NY, USA, 1998.
- [Ora92] Oracle. *PL/SQL user's guide and reference*. 1992.

## REFERÊNCIAS

- [Ora06] Oracle. *Oracle® Data Integrator: Improve Data Integration with Changed Data Capture*, 2006.
- [Ora07a] Oracle. *Oracle® Data Integrator API Reference*, 2007.
- [Ora07b] Oracle. *Oracle® Data Integrator: Knowledge Modules Developer's Guide*, 2007.
- [Ora07c] Oracle. *Oracle® Data Integrator User's Guide 10g Release 3*, 2007.
- [RG03] R. Ramakrishnan and J. Gehrke. *Database Management Systems*. McGraw-Hill Science/Engineering/Math, 2003.
- [Sca03] B. Scalzo. *Oracle DBA Guide to Data Warehousing and Star Schemas*. Prentice Hall PTR, 2003.
- [Sch08] K. Schlegel. Magic Quadrant for Business Intelligence Platforms, 1Q08, 2008.
- [UHM04] S. Urman, R. Hardman, and M. McLaughlin. *Oracle Database 10g PL/SQL Programming*. McGraw-Hill Osborne Media, 2004.
- [UW97] J.D. Ullman and J. Widom. *A first course in database systems*. Prentice-Hall, Inc. Upper Saddle River, NJ, USA, 1997.

# Anexo A

## Planeamento do projecto

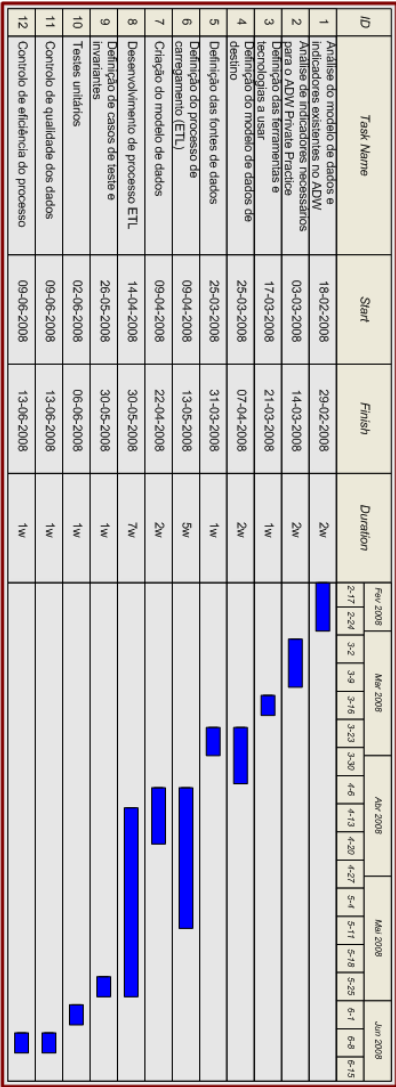


Figura A.1: Diagrama de Gantt do projecto.

## Planeamento do projecto

## Anexo B

# Organização da equipa de ADW

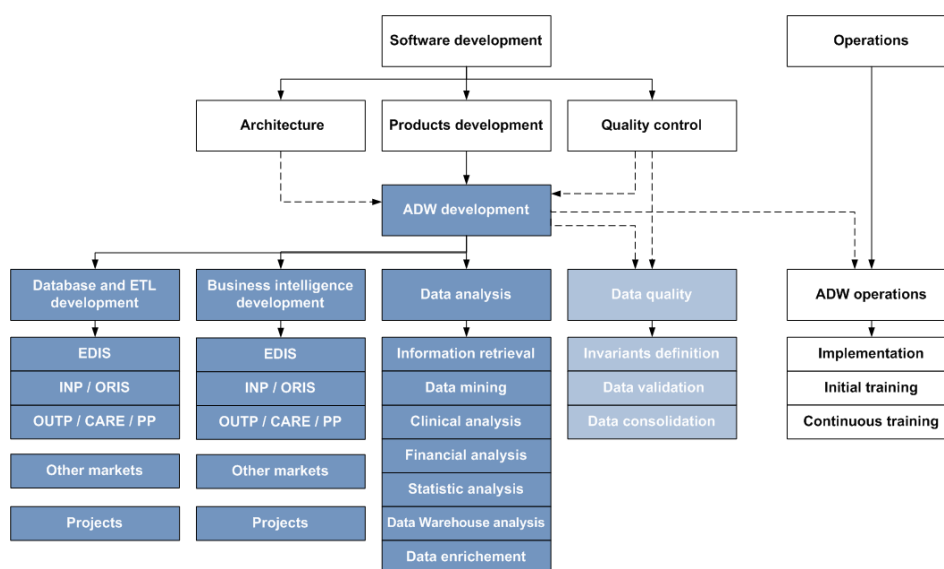


Figura B.1: Organização da equipa de ADW.