ASPECTOS DE CRIAÇÃO E CARGA DE UM AMBIENTE DE DATA WAREHOUSE

ANDRÉ FERNANDES DA COSTA
FELIPE CURVELLO ANCIÃES

Universidade Federal do Rio De Janeiro – UFRJ
Instituto de Matemática - IM
Departamento de Ciência da Computação
Projeto Final de Curso

Orientadora: Maria Luiza Machado Campos
Ph.D. em Ciência da Computação

RIO DE JANEIRO – RJ
MARÇO/2001

ASPECTOS DE CRIAÇÃO E CARGA DE UM AMBIENTE DE DATA WAREHOUSE

ANDRÉ FERNANDES DA COSTA
FELIPE CURVELLO ANCIÃES

Projeto Final de Curso submetido ao Departamento de Ciência da Computação do Instituto de Matemática da Universidade Federal do Rio de Janeiro como parte dos requisitos necessários para obtenção do grau de Bacharel em Informática.

Apresentado por:

_____

André Fernandes da Costa

_____

Felipe Curvello Anciães

Aprovado por:

_____

Profª. Maria Luiza Machado Campos, Ph.D.
(Presidente)

_____

Profª. Maria Claudia Reis Cavalcanti, M.Sc.

_____

Profª. Mônica Ferreira da Silva, M.Sc.

RIO DE JANEIRO – RJ
MARÇO/2001

RESUMO

COSTA, André F., ANCIÃES, Felipe C. **Aspectos de Criação e Carga de um Ambiente de Data Warehouse.**
Orientadora: Maria Luiza Machado Campos. Rio de Janeiro:UFRJ/IM/NCE, 2001. Diss.

Este estudo examina as principais questões envolvidas no processo de extração, transformação e carga dos dados e os desafios de povoar um ambiente de data warehouse (ADW) com dados relevantes e confiáveis. Também são abordados os aspectos relativos ao projeto físico deste ambiente, de forma que possa melhor suportar a implantação do mesmo. Ao final, utilizamos o estudo de caso de um modelo de data warehouse para a Universidade Federal do Rio de Janeiro, apresentado por Vânia de Jesus Araújo, em sua tese de mestrado, como um laboratório para a aplicação dos conceitos apresentados ao longo dos capítulos.

ABSTRACT

COSTA, André F., ANCIÃES, Felipe C. **Creation and Load Aspects of a Data Warehouse Environment.**
Orientadora: Maria Luiza Machado Campos. Rio de Janeiro:UFRJ/IM/NCE, 2001. Diss.

This work examines the principal questions that have to do with the extract, transformation and load processes of data and the challenges of populating a data warehouse with important and reliable data. It also covers the aspects concerning the physical project of such an environment, to better support the implementation of it. In the end, we utilize the case study of a data warehouse model for the Federal University of Rio de Janeiro, presented by Vânia de Jesus Araújo, in her mastering thesis, as a laboratory for the application of the concepts presented throughout the chapters.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ANEXOS

SUMÁRIO

CAPÍTULO 1

INTRODUÇÃO

Há algum tempo, as empresas buscam alternativas de se aumentar a utilidade dos dados presentes nos sistemas de informação. As organizações que conseguem extrair o máximo de informações sobre seus negócios a partir de seus dados, obtêm grande vantagem competitiva frente aos seus competidores, visto que elas conseguem maximizar o poder de decisão que está por trás dos mesmos.

Entretanto, o alto volume de dados aliado ao fato de eles estarem distribuídos entre os diversos sistemas da empresa exige um ambiente tecnológico capaz de lidar com tal complexidade. Para atender a esta necessidade surgiu a tecnologia de data warehousing.

A idéia-chave por trás da abordagem de data warehousing é a conversão de uma grande massa de dados em informações relevantes de uso estratégico para a empresa (BERSON, SMITH, 1997). O objetivo principal é suportar a tomada de decisões através da consolidação, conversão, transformação e integração dos dados operacionais, provendo uma visão consistente dos negócios da empresa.

1.1 Justificativa do Estudo

Para que um data warehouse (DW) consiga atingir os benefícios esperados, é preciso trabalhar os dados dos sistemas legados de forma a transformá-los em informações significativas.

Um data warehouse é tão bom quanto melhor forem os dados que ele contém. A fase de coleta, transformação e carga dos dados é a mais longa e a de maior risco em todo o processo de implementação de um data warehouse, podendo chegar a até 80% de tempo e custo de um projeto de DW (CAMPOS, FILHO, 1997).

Há inúmeras técnicas e abordagens que tratam do processo de extração, limpeza, transformação e carga dos dados, além de diversos produtos que se propõem a automatizar estas tarefas. No entanto, não existe uma solução única que suporte todo este processo.

1.2 Objetivo do Estudo

A finalidade do nosso trabalho é examinar as principais questões envolvidas no processo de extração, transformação e carga dos dados, e os desafios de povoar um data warehouse com dados relevantes e confiáveis, estabelecendo uma correlação com a tese "Modelagem Incremental no Ambiente de Data Warehouse", defesa de mestrado de Vânia Soares (SOARES, 1998). Abordaremos também aspectos relativos ao projeto físico deste ambiente, de forma que possa melhor suportar a implementação do data warehouse.

Parte deste estudo será aplicado no estudo de caso de implementação do Data Warehouse do Registro Acadêmico de Estudantes da Universidade Federal do Rio de Janeiro.

1.3 Organização do Trabalho

Este trabalho encontra-se organizado em seis capítulos.

O capítulo 2 apresenta as principais características de um ambiente de data warehouse (ADW), relacionando as arquiteturas e as metodologias existentes para a construção dos mesmos.

No capítulo 3 são discutidas importantes questões relacionadas com o processo de extração, limpeza, transformação e carga dos dados em um ADW, descrevendo os principais passos e técnicas empregadas por este processo.

O capítulo 4 abrange as questões relativas à definição de um modelo físico para o data warehouse, a estratégia de criação de agregados para se obter um melhor desempenho do ambiente, e ainda, a forma como planejar a infra-estrutura de um ambiente de data warehouse de modo a não limitar sua evolução inicial.

O capítulo 5 apresenta a aplicação das diretrizes em um estudo de caso no ambiente de registro acadêmico e resultados do vestibular. Este estudo consiste nos passos para implementação de dois DM: o DM Graduação e o DM Vestibular e a integração dos mesmos ao DW Universidade.

E, por fim, o capítulo 6 apresenta uma conclusão geral do trabalho, com considerações sobre o estudo e sugestões de trabalhos futuros.

Também fazem parte deste trabalho:

Anexo 1 – Dicionário de Dados referente ao Estudo de Caso Universidade;

Anexo 2 – *Scripts* de Extração, Transformação e Carga dos Dados do Estudo de Caso Universidade;

Anexo 3 – Apoio ao Estudo de Caso da Universidade.

CAPÍTULO 2

AMBIENTE DE DATA WAREHOUSE

O termo “data warehouse” designa um ambiente, e não um produto. Constitui uma arquitetura que provê informações de suporte à decisão que são difíceis de serem acessadas no ambiente operacional.

A tecnologia de data warehousing abrange um conjunto de tecnologias e componentes que se destinam a efetuar a integração dos bancos de dados operacionais em um ambiente que permita o uso estratégico dos dados.

Este capítulo se destina a apresentar e discutir as características deste ambiente, os diferentes tipos de arquitetura e as metodologias de construção de um ambiente de data warehouse.

2.1 Características de um Data Warehouse

O termo “data warehouse” foi popularizado por Bill Inmon. Ele define um data warehouse como sendo uma coleção de dados orientada por assuntos, integrada, variante no tempo, e não volátil, que tem por objetivo dar suporte aos processos de tomada de decisão. Esta definição é melhor explicada da seguinte forma (INMON, 1997):

- *Orientada por assuntos*: os dados de um data warehouse são organizados em torno de áreas específicas de uma empresa, como vendas, produtos, cobrança, etc.
- *Integrado*: o data warehouse integra dados com diferentes formatos provenientes de sistemas heterogêneos e os transforma em dados com uma representação única e consistente.
- *Variante no tempo*: os dados contidos em um data warehouse são relativos a um determinado momento ou intervalo de tempo, não sendo atualizáveis. A cada ocorrência de mudança, uma nova entrada é criada para identificá-la.
- *Não volátil*: Depois de trazidos para o ambiente de data warehouse, os dados, em geral, não sofrem mais modificações. Mudanças em um ambiente de data warehouse

ocorrem de forma controlada e planejada, ao invés dos ambientes OLTP onde as atualizações ocorrem continuamente. Esta volatilidade requer um trabalho considerável para assegurar integridade e consistência através de atividades de *rollback*, recuperação de falhas, *commits* e bloqueio – mecanismos desnecessários em um ambiente de data warehouse.

Um data warehouse pode ser visto como um ambiente com as seguintes características (BERSON, SMITH, 1997) (POE, KLAUER,BROBST, 1998):

- Centrado na figura de um repositório de dados que contém dados de sistemas heterogêneos e cuja finalidade é fornecer, através de análises, subsídios para o processo de tomada de decisão;
- Suporta um pequeno número de usuários, porém com interações relativamente longas;
- Orientado a consultas, que normalmente resultam em um grande conjunto de dados;
- Seu conteúdo é atualizado periodicamente (na maioria das vezes, adições);
- Contém poucas tabelas, porém grandes;
- Contém dados atuais e históricos que permitem uma perspectiva temporal sobre as informações;
- Visualização dos dados em diferentes níveis de sumarização.

2.1.1 Diferenças entre o Ambiente Operacional e o Ambiente de Data Warehouse

O ambiente operacional é composto por sistemas que têm como finalidade automatizar e suportar as principais atividades do dia-a-dia de uma empresa, e que geram um alto volume de transações – sistemas orientados a processos (*On-Line Transaction Processing* - OLTP). Por esta razão, os sistemas que compõem este ambiente devem ter alto desempenho e tempos de respostas baixos, visto que isto causa impacto diretamente na eficiência dos processos da empresa. Os dados operacionais são parte da infra-estrutura corporativa: são detalhados, atualizáveis e não-redundantes.

Por outro lado, o ambiente de data warehouse foi projetado para fornecer informações que apoiem o processo de tomada de decisão. Em geral, os dados informacionais são sumarizados e redundantes, suportando diferentes visões sobre eles, além de não serem atualizáveis.

O ambiente de data warehouse não difere do ambiente operacional somente quanto ao foco, mas também quanto ao escopo. Os dados transacionais são normalmente focados em áreas específicas, enquanto o ambiente de data warehouse precisa englobar grandes volumes de dados de diferentes áreas da empresa, e transformá-los em dados consistentes para poderem ser utilizados para análise.

A tabela abaixo lista as principais diferenças existentes entre o ambiente operacional e o ambiente de data warehouse (GOODYEAR *et al*, 1999).

Tópico/Função	OLTP	Data Warehouse
Conteúdo	Dados correntes, atômicos, isolados	Dados históricos, integrados, sumarizados
Organização dos Dados	Por sistema	Por assunto
Natureza dos Dados	Dinâmicos, com atualizações contínuas	Estáticos, com atualizações programadas
Estrutura dos Dados	Estruturados de forma a permitir uma leitura/escrita eficiente de registro em registro	Estruturado de forma a permitir acesso de leitura eficiente de grande volumes de dados
Tipo de Acesso	Grande volumes de inserções, atualizações e consultas	Grande volumes de registros sendo acessados por consultas complexas
Consultas e Índices	Otimização do BD de forma a permitir acesso eficiente a registros individuais	Otimização orientada para atender consultas que acessam uma grande quantidade de registros
Uso	Processamento altamente estruturado e de forma pré-definida	Processamento analítico, não estruturado, feito de acordo com a necessidade do usuário
Tempo de Resposta	Em segundos	De segundos a minutos

Tabela 2.1 – Ambiente Operacional x Ambiente de Data Warehouse.

Devido a estas diferenças, opta-se em separar os dados de caráter operacional daqueles que dão suporte à tomada de decisão. Assim, uma melhor funcionalidade e desempenho são obtidos para cada caso específico.

2.2 Princípios Direcionadores de um Ambiente de Data Warehouse

A forma como um ambiente de data warehouse é estruturado é determinada por decisões de como armazenar e distribuir os dados, assim como pelos modelos lógico e físico do data warehouse (GOODYEAR *et al*, 1999). Estas decisões devem estar baseadas nos requisitos dos usuários, que irão direcionar o desenho da arquitetura do data warehouse:

Granularidade de Dados: determina o grau de sumarização dos dados contidos no data warehouse. O nível de sumarização deve ser determinado pelos requisitos de negócios. Em um mesmo ambiente de data warehouse, podem existir diferentes níveis de granularidade, pois ao contrário dos dados detalhados, as visões dimensionais e os próprios agregados (analisados em detalhe no capítulo 4) podem se apresentar como dados já sumarizados. Em geral, quanto maior o nível de granularidade, maior será o número de acessos a estes dados, além de serem mais rápidos e mais eficientes.

Atualização e Tempo de Retenção dos Dados: este fator basicamente se relaciona a duas questões:

1) O quão atuais devem estar os dados de acordo com os requisitos de negócio?
2) Quanto tempo os dados devem estar armazenados antes de arquivá-los?

Estas questões são extremamente importantes pois impactam diretamente na capacidade dos dispositivos de armazenamento, no tráfego da rede, e além disso, nos processos de extração, transformação e carga dos dados, que tomarão mais tempo à medida que a granularidade for menor, diminuindo assim a janela de disponibilidade do ambiente de data warehouse.

Disponibilidade: a disponibilidade do ambiente de data warehouse deve ser determinada pelos requisitos dos usuários. Isto pode afetar diretamente na forma como a arquitetura técnica será projetada, tornando os custos mais altos conforme a necessidade de

implantar *hardware* e *software* para garantí-la (por exemplo, replicação e espelhamento de dados).

Escalabilidade: este é um ponto ao qual se deve ter bastante atenção, pois um ADW se encontra em crescimento contínuo. Esta preocupação deve direcionar o projeto de implementação da infra-estrutura técnica, como veremos adiante no capítulo 4.

2.3 Arquitetura do Ambiente de Data Warehouse

Para que um ambiente de data warehouse represente mais que um simples conjunto de ferramentas e tecnologias, e que uma vez implementado, evolua em conjunto com os negócios da empresa – mantendo a promessa de fornecer informações úteis a seus usuários – é essencial que se tenha por trás dele uma arquitetura de sistemas bem definida. Um desenho conceitual do ambiente de data warehouse deve ser especificado , assim como o *hardware* e o *software* que o suportam, e os padrões e serviços que o fazem funcionar. A figura 2.1 apresenta uma visão simplificada de uma arquitetura de data warehouse.

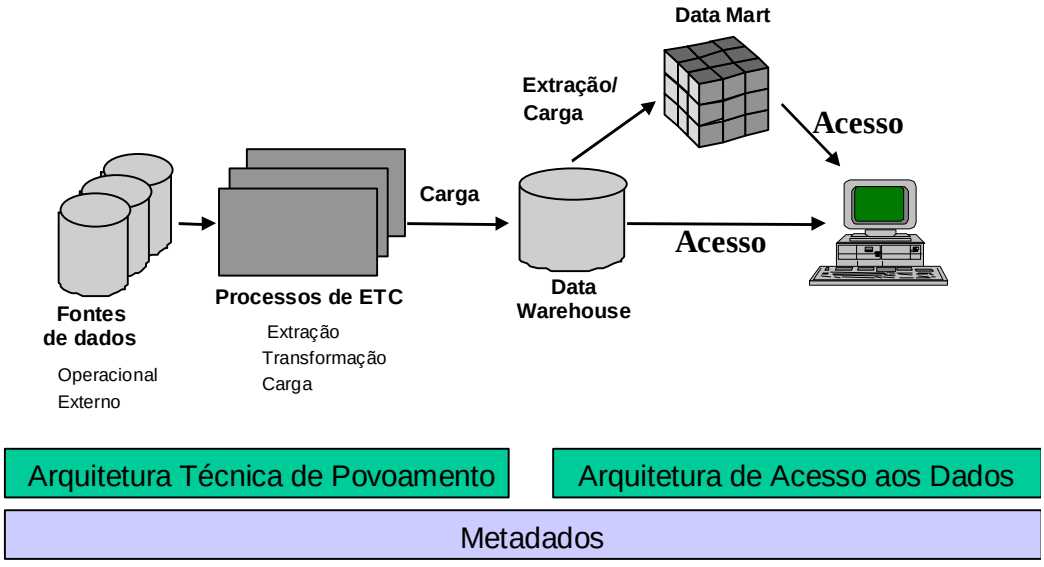

Figura 2.1 – Visão simplificada de uma arquitetura de Data Warehouse.

2.3.1 Componentes

A arquitetura de um ambiente de data warehouse abrange, além de estruturas de armazenamento, mecanismos de integração, comunicação, processamento e apresentação da informação para o usuário final. Pode-se dizer que ela é composta pelas seguintes partes:

- Arquitetura de Dados;
- Arquitetura Técnica;
- Administração e Gerenciamento do Data Warehouse;
- Metadados.

Arquitetura de Dados

A arquitetura de dados descreve o conteúdo do data warehouse – os dados que são importantes para o negócio, as estruturas de armazenamento que compõem o ambiente do data warehouse e as fontes que as alimentam. Também inclui os modelos de dados lógicos e físicos, agregações e hierarquias, tudo isto baseado nos requisitos levantados. Além disso, define a granularidade dos dados, o volume e a distribuição dos dados no ambiente de data warehouse (KIMBALL *et al*, 1998).

Arquitetura Técnica

A arquitetura técnica abrange os processos e as ferramentas que atuam sobre os dados. É ela que se preocupa com a forma com a qual os dados serão extraídos da fonte, como eles serão tratados de modo a atender os requisitos do negócio, e como fazer para que eles se tornem acessíveis para o usuário. É responsável pelo gerenciamento das atividades que constróem e mantêm as informações do data warehouse (KIMBALL *et al*, 1998).

A arquitetura técnica é composta por duas áreas distintas que merecem ser consideradas independentemente: o *back room* e o *front room*. O *back room* abrange os processos de carga inicial e das atualizações periódicas do DW, sendo assim, responsáveis pela extração dos dados de múltiplos sistemas operativos e fontes externas,

pela limpeza, transformação e integração destes dados. O *front room* é a face pública do data warehouse. É responsável por disponibilizar os dados para o usuário final, contendo as ferramentas que este utiliza no dia-a-dia. Também envolve o *hardware* e *software* utilizados para a elaboração de relatórios, pesquisas informativas, análise e "data mining".

Administração e Gerenciamento do Data Warehouse

É responsável por toda a infra-estrutura do ambiente de data warehouse (*hardware*, rede, etc.) e pelo gerenciamento dos serviços que contribuem para manter o data warehouse atualizado e consistente.

De um modo geral, o gerenciamento de um data warehouse inclui as seguintes atividades (BERSON, SMITH, 1997) :

- Segurança;
- Gerenciamento das atualizações de diferentes fontes;
- Checagem da qualidade dos dados;
- Garantia das atualizações dos metadados;
- Monitoramento da performance do data warehouse;
- Expurgo de dados antigos;
- Coordenação das atividades de replicação e distribuição dos dados;
- *Backup* e *recovery*;
- Dimensionamento do *hardware.*

Metadados

Os metadados consistem em informações sobre dados que compõem o data warehouse. Eles são extremamente importantes dentro do ambiente de data warehouse, pois representam uma visão integrada das bases de dados que fazem parte deste ambiente. Eles são utilizados para construir, manter, gerenciar e utilizar o data warehouse.

2.3.2 Repositórios de Dados

Embora, por sua própria natureza, o ambiente de data warehouse seja um grande centralizador de dados, existem distintas formas de se armazenar e distribuir os dados de modo a atender os requisitos do usuário de desempenho e disponibilidade estabelecidos. O tipo de arquitetura escolhida define a utilização ou não dos repositórios apresentados a seguir:

Data Warehouse

Consiste na figura central do ambiente de data warehouse. É uma grande base de dados que integra dados selecionados e depurados, provenientes de múltiplos sistemas operativos e fontes externas, e que visa apoiar o processo de tomada de decisões. Em geral, contém informações sumarizadas de diferentes áreas da empresa, que são mantidas por longos períodos de tempo, para fins analíticos.

Não deve ser confundido com o banco de dados corporativo da empresa, pois as informações que contém possuem um nível de granularidade e até mesmo formatos diferentes dos existentes no ambiente operacional, visto que tem por objetivo atender ao uso estratégico dos dados.

O data warehouse deve ser fisicamente otimizado para lidar com grande volume de dados e consultas complexas, e menos indexado e normalizado do que os repositórios de dados encontrados no ambiente operacional.

***Operational Data Store* – ODS**

Armazena dados do ambiente operacional com o intuito de permitir uma visão atual e integrada dos mesmos para fins de análises em um curto espaço de tempo, antes de sua atualização no data warehouse. Em sua proposta inicial, o ODS armazenaria dados temporariamente (p.ex., 24 horas), sendo atualizado continuamente de acordo com o ambiente operacional. Enquanto isto faz com que o ODS mantenha sua quantidade de dados em um nível mínimo, torna as atualizações dos dados extremamente difíceis (INMON, IMHOFF, BATTAS, 1996).

Pelo fato de já conter os dados do ambiente operativo integrados, o ODS pode ser utilizado para agilizar o processo de consolidação, proporcionando um melhor desempenho na fase de atualização dos dados no data warehouse.

Se considerarmos o ODS proposto por Inmon como sendo uma necessidade de integração dos dados do ambiente operativo, podemos compará-lo com um data warehouse da seguinte forma:

- Assim como o data warehouse, o ODS também é orientado por assuntos;
- O ODS é integrado, da mesma forma que um data warehouse;

Entretanto,

- ODS é volátil, enquanto o data warehouse não é;
- O ODS contém os dados mais atualizados, enquanto o data warehouse contém dados históricos e atualizados;
- Ao contrário do data warehouse, o ODS só contém dados detalhados, e não possui agregados e dados sumarizados.

A figura abaixo indica as principais diferenças existentes entre o ODS volátil e um data warehouse.

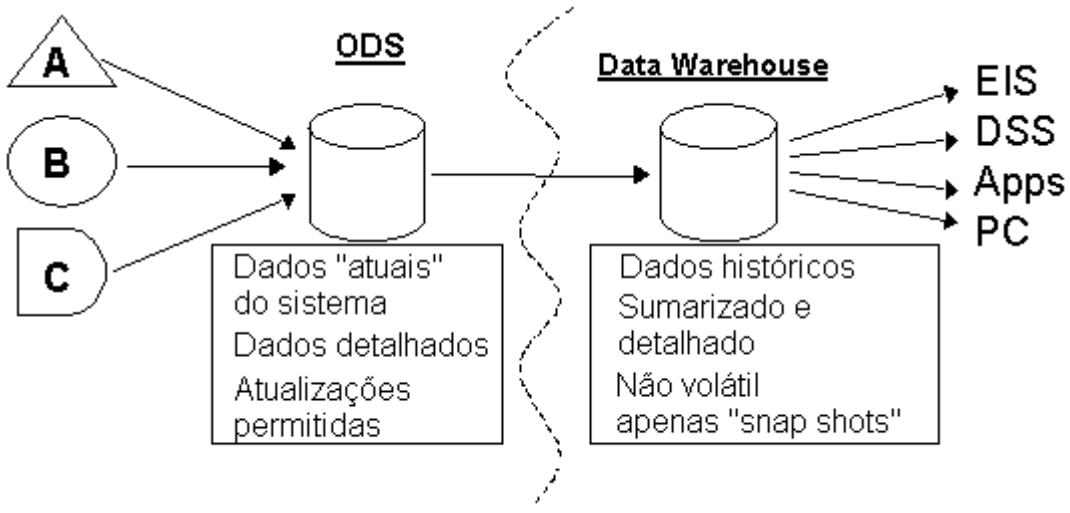

Figura 2.2 – Diferenças entre ODS e Data Warehouse.

Com a evolução das tecnologias que suportam o ambiente de data warehouse (*hardware* e *software*), este além de conter dados sumarizados, passou a contemplar também dados atômicos. De uma certa forma, o ambiente de data warehouse absorveu o propósito do ODS ao armazenar dados detalhados. Sendo assim, Kimball redefiniu o ODS como sendo uma estrutura de armazenamento de dados detalhados, não-volátil, orientada a assuntos e que suporta os sistemas do ambiente operacional com dados integrados. De qualquer forma, o uso de dados detalhados para tomada de decisões deveria se basear no nível de granularidade mais baixo suportado pelo data warehouse (KIMBALL *et al*, 1998).

Em princípio, o ambiente de data warehouse não requer a figura de um ODS. A sua criação deve ser uma decisão do projeto, considerando os custos e o tempo necessários para construí-lo.

Data Mart (DM)

O data mart é uma base de dados que contém dados operacionais específicos de uma determinada área ou assunto, e que apoiam a tomada de decisão de um grupo particular de usuários. Logo, é comum que uma empresa possua DMs específicos para determinados departamentos e setores, como mostra a figura 2.3.

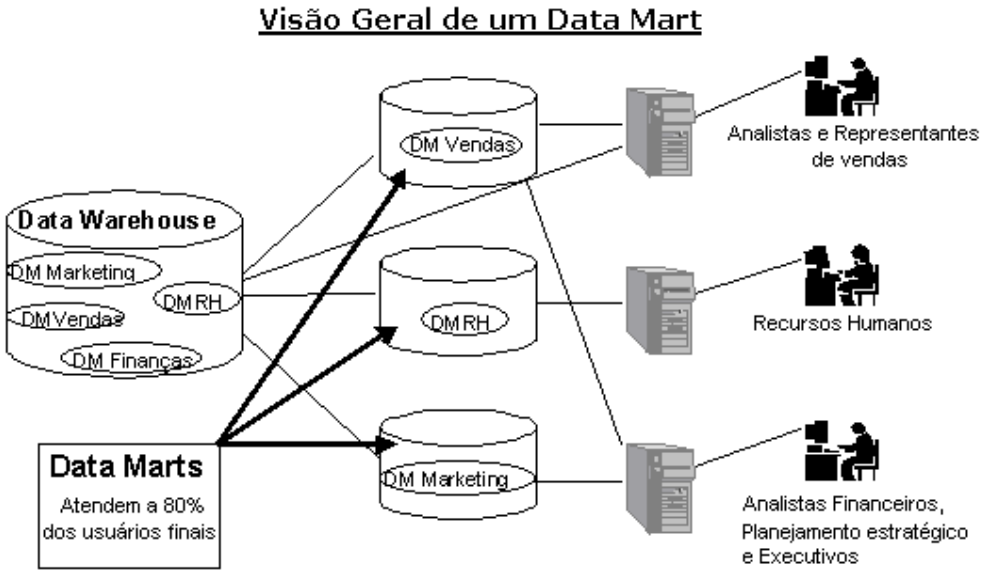

Figura 2.3 – Visão Geral de um Data Mart

Os DMs apresentam armazenamento de dados altamente indexado, para permitir acesso rápido para a realização de análises por parte do usuário. É comum também armazenarem os dados em Sistemas Gerenciadores de Bancos de Dados Multidimensionais (SGBDM), pois estes apresentam boa flexibilidade de análise, embora não sejam recomendados para o armazenamento de grandes volumes de dados.

A forma mais eficaz de se alimentar os DMs é através de uma fonte única de dados – o data warehouse – que permite uma visão consistente dos mesmos por toda a empresa. A carga dos DMs realizada diretamente dos sistemas do ambiente operacional introduz o risco de propagação de visões inconsistentes dos mesmos dados pela empresa.

Cada data mart, de um modo geral, possui uma infra-estrutura própria – *hardware*, *software*, dados e aplicações específicas. Isto torna o controle e a gerência dos dados uma tarefa extremamente árdua, e é uma das principais razões para que o data warehouse funcione como um grande centralizador dos dados que estão distribuídos pelos diferentes DMs.

Critério	ODS	Data Warehouse	Data Mart
Função	Suporte operacional a nível de departamento / área	Suporte à tomada de decisão corporativa, departamental ou de determinada área do negócio	Suporte à tomada de decisão departamental ou área de negócio
Fonte dos Dados	Sistemas operativos e fontes externas	Sistemas operativos e fontes externas	Sistemas operativos e fontes externas ou dados provenientes de um data warehouse
Esquemas	Normalizados	Dimensional ou relacional dependendo do uso	Dimensional para aumentar a performance
Escopo	Limitado	Amplo	Limitado
Tempo	Dados atuais e sincronizados com os sistemas operativos	Dados históricos, para permitir a realização de análises e planejamento estratégico	Dados históricos, para permitir a realização de análises e planejamento estratégico
Volatilidade/ Frequência de Atualização	Alta; os dados freqüentemente estão sendo atualizados para refletir as mudanças no ambiente operativo	Baixa/Média; os dados são estáticos e as mudanças dependem da freqüência de atualização (em geral, inserções)	Baixa; os dados são estáticos e as mudanças só ocorrem quando atualizações nos dados históricos são necessárias
Agregação dos Dados	Nível baixo; em geral, dados detalhados	Médio; alguns dados sumarizados são requeridos para aumentar performance e padronizar as análises. Níveis detalhados de dados transacionais históricos	Médio; alguns dados sumarizados são requeridos para aumentar performance e padronizar as análises. Somente contém dados detalhados necessários para análises específicas

Tabela 2.2 – Comparação entre ODS, Data Warehouse e Data Marts (GOODYEAR *et al*, 1999).

2.3.3 Tipos de Arquitetura

Atualmente, há diversas abordagens para a estruturação de um ambiente de data warehouse, devendo ser escolhida aquela que melhor se adequa às necessidades do negócio. Variações das arquiteturas foram surgindo, de modo a acomodar a integração de todos os componentes de um ADW. A escolha da arquitetura é fator primordial para o sucesso da implementação de um ADW. Hoje, considera-se que os problemas do ADW estão mais relacionados com a arquitetura selecionada do que propriamente com a tecnologia disponível (MELO, 1997). Os desenvolvedores deste ambiente devem se preocupar em como integrar o DW às diversas fontes heterogêneas e externas, aos DMs, ODS, aplicações servidoras, "WEB" e "data mining", entre outros tipos de ferramentas disponíveis (FIRESTONE, 1998).

Arquitetura "Top-Down"

Introduzida por Bill Inmon (INMON, HACKATHORN, 1997), é o primeiro tipo de arquitetura de um ambiente de data warehouse, sendo hoje a mais conhecida e considerada como arquitetura padrão. A figura 2.4 apresenta este tipo de arquitetura.

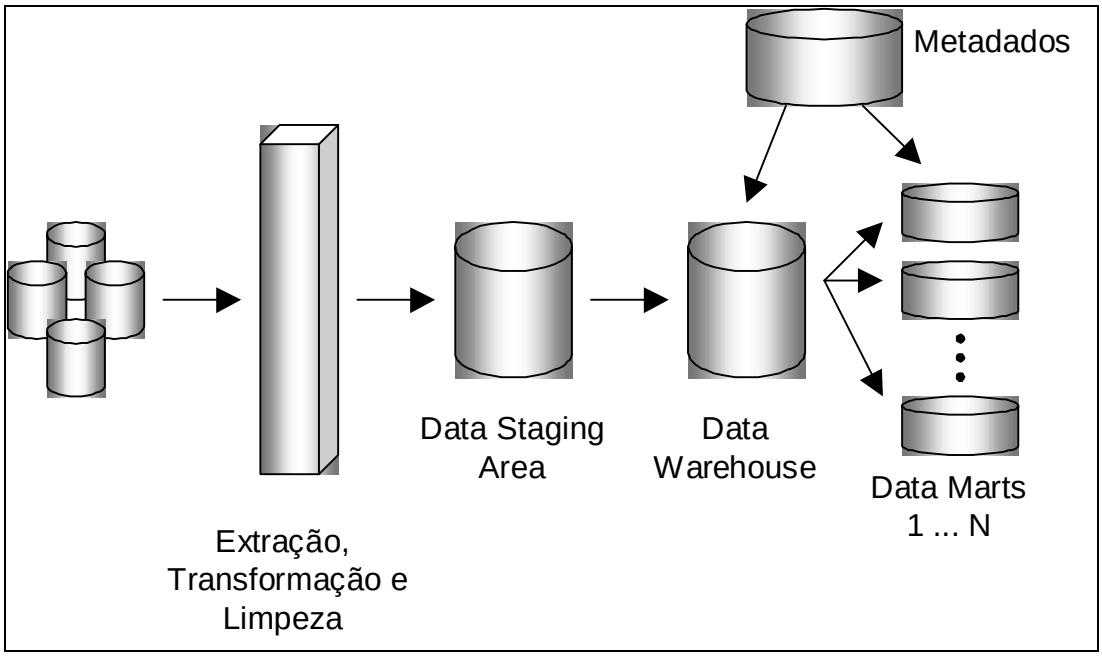

Figura 2.4 – Arquitetura Padrão

Esta arquitetura se baseia na extração, transformação e carga dos dados provenientes do ambiente operacional e de fontes externas para uma área intermediária de *staging* dos dados. Em seguida, os dados e os respectivos metadados são carregados no data warehouse propriamente dito. Desta forma, o DW consiste em uma camada de dados detalhados e históricos. Somente a partir destes dados, que os data marts serão carregados com os mesmos já sumarizados.

Enquanto que os data marts são modelados através do esquema-estrela, os dados detalhados contidos no data warehouse são armazenados em um modelo ER típico de um ambiente operacional. No entanto, como o data warehouse consiste em um ambiente puramente de consultas, muitas das restrições existentes no ambiente operacional podem ser relaxadas.

Esta arquitetura garante a consistência entre o data warehouse e os data marts, pois estes são construídos com base em um subconjunto dos dados mantidos pelo DW. A grande desvantagem deste tipo de arquitetura é o longo tempo demandado para a sua implementação, fazendo com que os altos custos e os riscos associados sejam fatores inibidores para sua adoção.

Arquitetura “Bottom-Up”

A idéia central por trás desta arquitetura é a da construção de um ambiente de data warehouse de forma incremental através do desenvolvimento de data marts independentes. Esta arquitetura não requer uma área comum de *staging* de dados, podendo cada data mart possuir sua própria área. Os processos ETL de cada data mart são totalmente disjuntos, podendo não existir sequer uma padronização das ferramentas.

Como o data warehouse é gerado a partir do conjunto de data marts existentes, estes devem possuir todas as representações de dados do ambiente, contendo não somente dados sumarizados, mas também dados históricos detalhados.

Do ponto de vista de integração, este tipo de arquitetura possui uma diferença crucial para a arquitetura “top-down”: não existe padronização dos metadados dos data marts. Este fator faz com que a arquitetura “bottom-up”, embora obtenha sucesso inicial devido a sua rápida implementação e retorno, se torne inviável ao longo do tempo, pois

ela falha justamente no ponto ao qual se propõe: construir o ambiente de data warehouse a partir dos data marts – o que é muito difícil pois não há metadados compartilhados (FIRESTONE, 1998). A arquitetura “bottom-up” está representada na figura 2.5.

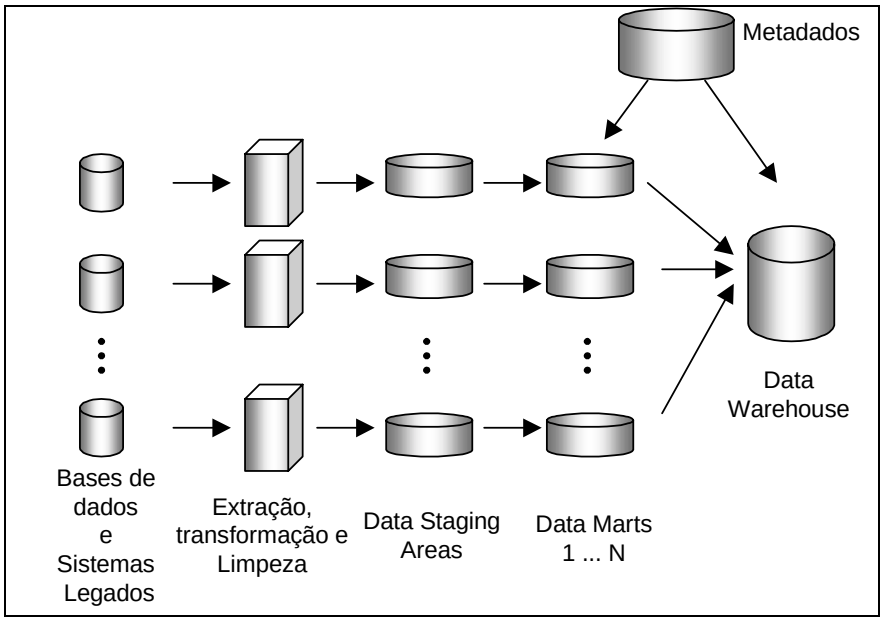

Figura 2.5 – Arquitetura Bottom Up

Novas abordagens surgiram como variações da arquitetura “bottom-up”, visando obter uma melhor integração dos components do ADW, assim como alcançar a consistência dos metadados. A seguir, listamos alguns exemplos destas novas abordagens:

Arquitetura EDMA (“Enterprise Data Mart Architecture”)

Esta arquitetura, representada na figura 2.6, evoluiu da idéia da arquitetura “bottom-up” - de construção incremental de um data warehouse – porém, através da utilização de um “framework” compartilhado para a implementação dos data marts (FIRESTONE, 1998).

Este “framework” contém áreas de assunto da empresa, dimensões comuns, métricas, regras de negócio e fontes de dados. Seu principal objetivo é garantir a

padronização dos metadados utilizados, de modo a permitir a implementação incremental do ambiente de data warehouse, sem que haja inconsistências.

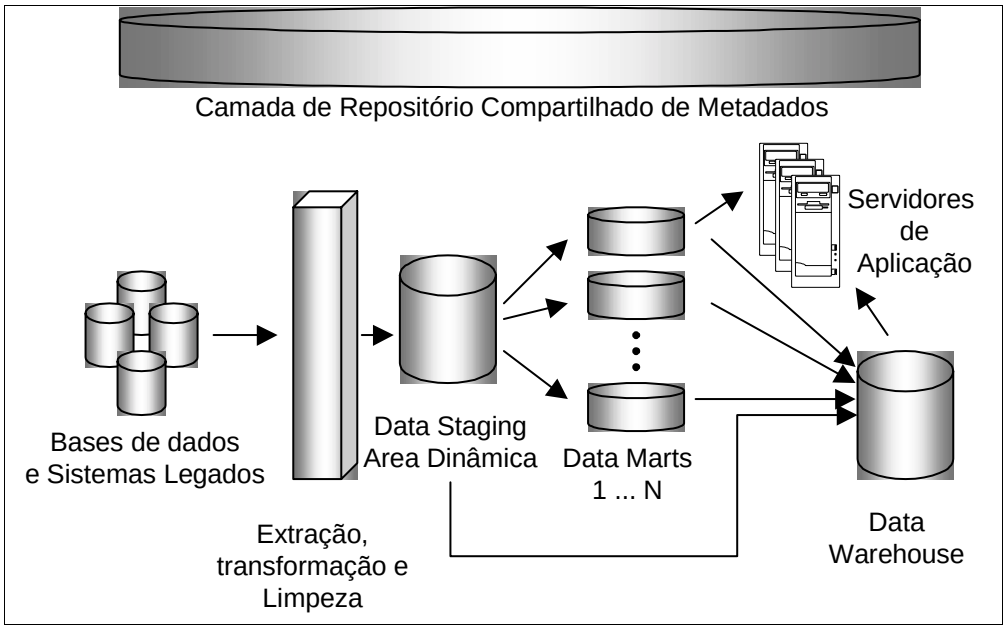

Figura 2.6 – Arquitetura EDMA

O “framework” é implementado através da figura de dois repositórios de dados:

- uma área de *staging* de dados comum chamada de DDS (“Dynamic Data Store”), para onde os dados são extraídos, transformados e preparados para as cargas nos data marts. Seu dinamismo está no fato dos dados que contém estarem em constante mudança;
- e um repositório global de metadados, que permite manter a consistência semântica dos dados.

Esta arquitetura representou um grande avanço na questão de administração dos metadados de um ADW.

Arquitetura DS/DM (“Data Stage/Data Mart Architecture”)

A idéia básica desta arquitetura é a mesma implementada pela arquitetura EDMA, porém com uma exceção primordial: o data warehouse não é fisicamente implementado, sendo substituído por uma conjunção lógica de data marts (FIRESTONE, 1998). No entanto, é difícil obter uma visão mais ampla dos dados da empresa, com propriedades globais, visto que os data marts são construídos com base em um escopo delimitado por uma determinada área de interesse. A figura 2.7 ilustra esta arquitetura.

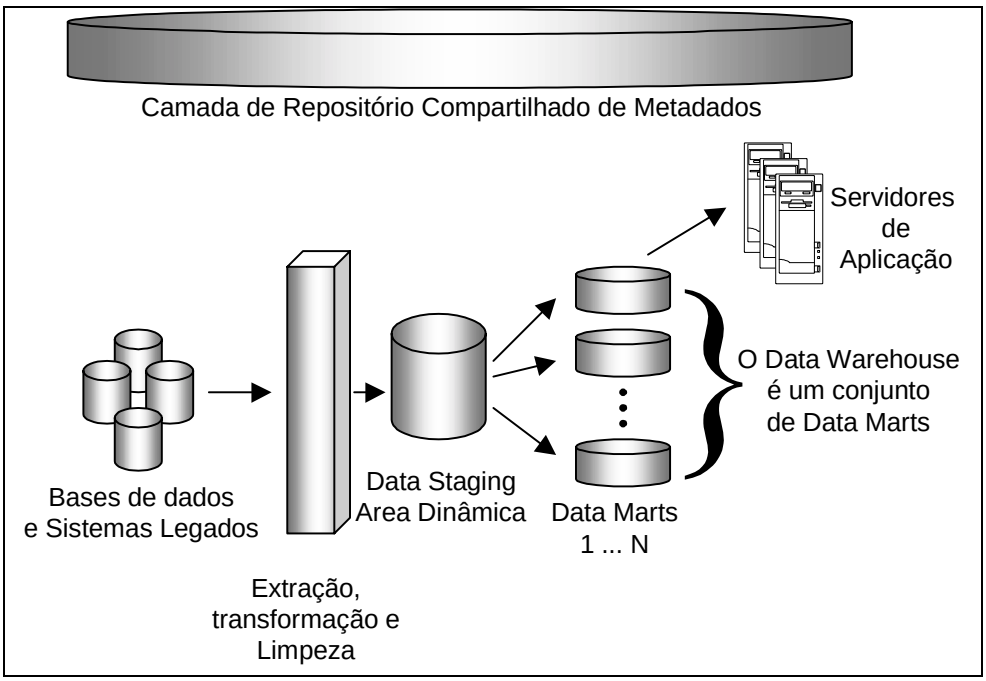

Figura 2.7 – Arquitetura DS/DM

Arquitetura DDW/DM (“Distributed Data Warehouse/Data Mart Architecture”)

Esta arquitetura também é similar à arquitetura EDMA. Assim como esta, também é composta por uma área de *staging* dinâmica e por um repositório de metadados comum (FIRESTONE, 1998). Porém, existem duas características que as distinguem:

- provê uma camada de banco de dados intermediária que realiza o mapeamento do modelo lógico com as tabelas físicas dos diversos data marts;
- permite que consultas sejam realizadas através desta camada sobre os dados contidos nos data marts e no data warehouse.

Estes dois fatores possibilitam a distribuição dos repositórios de dados do ambiente de data warehouse de foma transparente para os usuários.

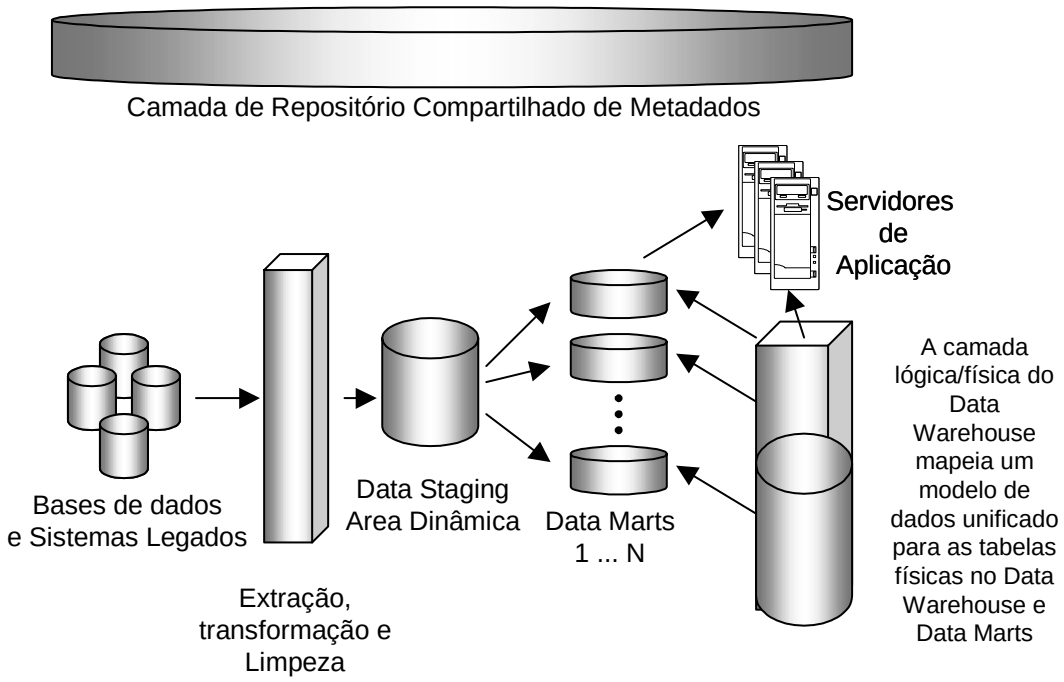

Figura 2.8 – Arquitetura DDW/DM

Arquitetura Intermediária

Embora tenham surgido novas propostas para a arquitetura de um ambiente de data warehouse, as arquiteturas “top-down” e “bottom-up” continuam sendo as mais utilizadas nos projetos de desenvolvimento de ADW. Com base nas características destas duas arquiteturas, Vânia Soares em sua tese de mestrado (SOARES, 1998) propôs uma arquitetura intermediária: nesta abordagem, realiza-se a modelagem de dados do data warehouse, e após isto, implementam-se partes deste modelo, ou seja, cada data mart construído é integrado ao modelo físico do DW. Como o modelo de

dados é único, possibilita-se o mapeamento e o controle dos dados, garantindo desta forma a consistência das informações. Em termos estruturais, esta arquitetura não difere em nada da arquitetura “Bottom-Up” representada pela figura 2.5.

CAPÍTULO 3

POVOAMENTO DE UM AMBIENTE DE DATA WAREHOUSE

Mais do que a arquitetura definida para o ADW, o sucesso de um data warehouse depende da qualidade dos dados que ele contém. Sendo assim, o processo de integração de dados de múltiplas fontes e sua transformação em informações consistentes e de qualidade para permitir o acesso por parte do usuário final é de vital importância para o êxito do ambiente de data warehouse. De acordo com especialistas, este processo demanda cerca de 80% do esforço de um projeto de implementação de um ambiente de data warehouse e, em muitos casos, ocupa muito mais tempo que o previsto (CAMPOS, FILHO, 1997).

Este capítulo tem o propósito de discutir todo o processo de extração, transformação e carga dos dados no ambiente de data warehouse, descrevendo os passos e as técnicas utilizadas por este processo, e levantando as principais questões que o envolvem. A figura 3.1 ilustra os passos específicos da etapa de povoamento de um data warehouse.

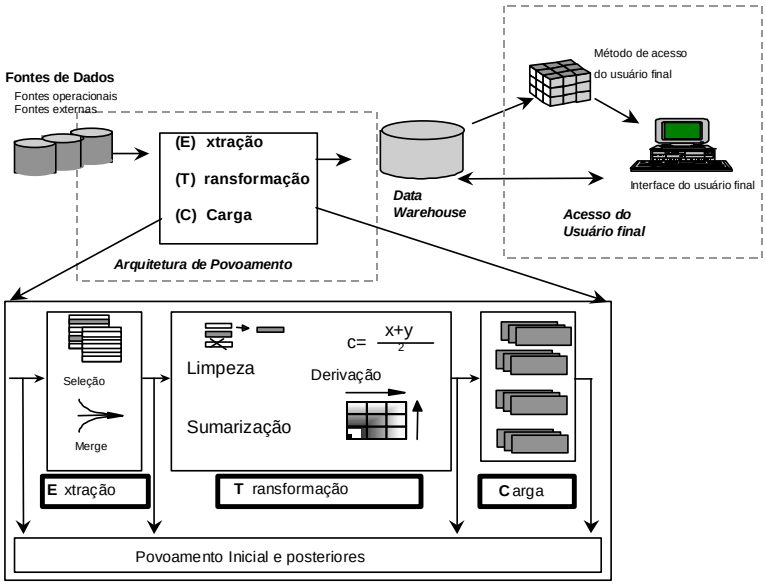

Figura 3.1 – Etapa de Povoamento de um Data Warehouse (GOODYEAR *et al*, 1999).

3.1 Etapas do Processo de Povoamento de um Data Warehouse

3.1.1 Extração

Esta etapa é responsável por identificar as melhores fontes de dados, e por obter os dados necessários destas fontes. Os dados a serem extraídos devem ser selecionados segundo o modelo de dados gerado pela etapa de modelagem do data warehouse. Isto faz com que a extração dos dados seja muito dependente da qualidade com a qual a modelagem foi realizada. Se o modelo de dados for de baixa qualidade, haverá um esforço adicional para mapear os requerimentos de dados com os sistemas existentes no ambiente operacional. Um outro fator que pode tornar esta etapa ainda mais complexa é o fato dos sistemas-fonte estarem disponibilizados em diferentes plataformas e tecnologias, o que demandará em alguns casos modos de extração diferenciados.

Os seguintes princípios podem ser citados para o sucesso da etapa de extração de dados (GOODYEAR *et al*, 1999):

- *Os dados devem ser precisos.* Se um departamento mantiver diferentes versões de dados para um mesmo período de tempo, devem ser escolhidos os dados mais apropriados para a extração;
- *Os dados devem ser os mais recentes.* Isto garante que o DW contenha dados sempre atualizados (pelo menos, o mais próximo possível disto);
- *Os dados devem estar fechados*. É recomendável que se espere para realizar a carga dos dados após o fechamento do dia no sistema-fonte. Para cargas mensais, o mês deve estar fechado antes que os dados sejam extraídos;
- *Os dados devem estar completos.* Se por exemplo, um sistema realiza análises sobre todos os registros e um outro sistema apenas sobre um subconjunto dos mesmos, esta diferença deve ser considerada no momento da extração;
- *Os dados devem estar o mais próximo possível de sua origem.* Dados que trafegam por diferentes sistemas dentro da empresa antes de serem extraídos para o DW possuem uma maior probabilidade de conter discrepâncias do que aqueles que são extraídos diretamente do sistema no qual se originam.

Diferentes técnicas de extração de dados podem ser utilizadas em um ambiente de data warehouse:

3.1.1.1 Extração Total

Os dados são extraídos inteiramente de suas fontes; as etapas de transformação e carga determinarão que dados serão utilizados. Existem duas formas de implementação:

- Carga incremental: Todos os dados são extraídos, porém a carga só é realizada para uma parcela dos mesmos. Uma cópia dos dados da última extração é comparada com os dados correntes para determinar as mudanças que ocorreram desde a última extração. Somente os dados que sofreram mudanças serão transformados e carregados no data warehouse.
- Carga total: Todos os dados são extraídos das fontes de dados, transformados e carregados, desconsiderando se ocorreram mudanças com os registros. Um exemplo de quando isto ocorre é durante a carga inicial de um data warehouse.

Critério	Carga Incremental	Carga Total
Tempo de Processamento	Mais lento, pois envolve comparações de registros	Mais rápido, pois não precisa comparar registros
Tempo de Transferência Para o DW	O tráfego pode ser reduzido se a comparação ocorrer antes da transferência	A transferência completa de dados pela rede pode ser altamente custosa para grandes volumes de dados
Complexidade	Requer mecanismos que implementem comparações eficientes	Não necessita de comparações
Desempenho da Carga	Somente os dados que mudaram são carregados	É impactado com as deleções/ atualizações de registros redundantes

Tabela 3.1 – Comparação entre as Abordagens de Extração Total dos Dados

3.1.1.2 Extração Parcial

Somente as transações que sofreram atualizações, foram inseridas ou excluídas das fontes de dados desde a última extração é que serão extraídas para o data warehouse, para que as etapas de transformação e carga possam tratá-las de modo diferente. A identificação destas mudanças pode ser feita com base no *log* de transações, ou em cima de *timestamps* ou *flags* indicadores de mudanças.

A tabela abaixo lista as principais diferenças existentes entre estas duas abordagens de extração.

Critério	Extração Total	Extração Parcial
Flexibilidade de Implementação	Não depende de como os dados são armazenados e mantidos	Depende fortemente da forma como os dados estão armazenados e mantidos
Impacto no Desempenho do Ambiente Operativo durante a Extração	Pode degradar o desempenho do ambiente operativo durante extensos períodos de extração	Em geral, a extração somente dos dados que mudaram pode ser realizada de forma relativamente rápida
Redundância	Informações que já estão no ADW também são extraídas, causando uma grande perda de tempo para um grande volume de dados	Extrai somente os dados que sofreram mudanças
Impacto no Tráfego da Rede	A largura de banda exigida é proporcional ao volume de dados a ser extraído	Não causa impacto na rede, pois só os dados que mudaram são extraídos
Impacto no Desempenho do ADW durante a Carga	Registros que já estavam no ambiente são reprocessados e reinseridos	Somente os registros que mudaram precisam ser carregados
Aplicabilidade	Aplicável para tabelas pequenas e de referência	Aplicável para grandes tabelas

Tabela 3.2 – Comparação entra Extração Total e Extração Parcial

Nas abordagens onde se mantêm um data warehouse corporativo e data marts departamentais, existe a necessidade de se estabelecer uma estratégia para coordenar a entrega de novos dados para todos os bancos. Portanto, é preciso considerar a incorporação de um servidor de replicação na arquitetura de povoamento do ambiente de data warehouse. Este servidor consiste em uma aplicação sofisticada que seleciona e particiona os dados para distribuí-los para cada data mart, aplicando restrições de segurança, replicando dados para os locais adequados e gravando em *log* todas as transmissões (GOODYEAR *et al*, 1999).

3.1.2 Transformação

Esta etapa é extremamente importante, pois através dela, os dados que foram extraídos e trazidos para o ambiente de data warehouse serão tratados de modo a se tornarem informações úteis para o usuário e relevantes para o negócio.

O fato de que o data warehouse é composto por dados de diferentes sistemas, cada um com um ambiente específico e com um propósito particular, faz com que esta etapa se torne ainda mais complexa.

Durante o processo de transformação dos dados, é fundamental que exista um gerenciamento de metadados eficiente que suporte este processo, armazenando as regras de negócio que garantam um entendimento correto e consistente dos dados.

As seguintes transformações provavelmente constarão na maioria dos projetos de implementação de um data warehouse (GOODYEAR *et al*, 1999):

- Integração
- Limpeza e Validação
- Derivação e Sumarização
- Atualizações do histórico
- Ordenação

3.1.2.1 Integração

Através deste passo, realiza-se a combinação dos dados provenientes de diferentes sistemas do ambiente operacional, e o mapeamento destes dados entre os sistemas fontes e o data warehouse. Cada dado extraído para o ambiente de data warehouse deve ser trazido de um determinado sistema e para um lugar específico, e deve ser formatado de modo a atender ao padrão definido para o data warehouse.

3.1.2.2 Limpeza e Validação

Este é um dos passos mais críticos do processo de transformação dos dados, pois ele é o principal responsável pela qualidade do dado que está contido no ambiente de data warehouse, o que constitui um fator crítico de sucesso deste ambiente. Se este passo for relegado a segundo plano ou mesmo desconsiderado, o resultado poderá ser um ambiente de data warehouse inconsistente e até mesmo inútil (HUFFORD, 1999).

É utopia acreditar que os dados provenientes dos sistemas do ambiente operacional são dados perfeitos e consistentes, e que não precisam ser validados (KIMBALL *et al*, 1998). Existem muitas “sujeiras” nos dados dos sistemas legados pelo simples fato de que elas não afetam em nada a funcionalidade destes sistemas, sendo portanto praticamente inúteis quaisquer esforços para eliminá-las em seu ambiente de

origem.

Cada implementação deve definir os níveis de qualidade de dados aceitáveis para seus ADWs. A seguir, estão listadas as principais características de qualidade dos dados (KIMBALL *et al*, 1998):

- *Consistência*: Os dados dentro de um data warehouse devem ser totalmente consistentes. Eles não podem, em hipótese alguma, apresentar contradições. Por exemplo, os agregados devem ser consistentes com os dados detalhados relacionados;
- *Unicidade*: Não podem ocorrer duplicidades dentro de um ADW. Se dois elementos são os mesmos, eles devem possuir a mesma chave. Por exemplo, UFRJ e U.F.R.J. representam a mesma universidade;
- *Completude*: Os dados contidos dentro de um data warehouse representam o conjunto inteiro de dados relevantes, e que está de acordo com o escopo definido. Se estamos analisando os dados de notas de todos os alunos da universidade, os alunos de todos os cursos devem estar contidos nesta análise, não podem faltar, por exemplo, os dados dos alunos de Informática;
- *Precisão*: Os dados dentro de um data warehouse devem refletir os dados dos sistemas de onde foram extraídos. Se por exemplo, um determinado campo do tipo número é diferente do que antes era representado, deve haver uma explicação como metadado que explique o porquê desta diferença.

Em resumo, o dado dentro de um ambiente de data warehouse deve ser " a verdade, a mais verdadeira verdade, e nada mais que a verdade" (KIMBALL *et al*, 1998).

As ações de limpeza dos dados podem ser categorizadas em três tipos: sintáticas, estruturais e semânticas.

A) Sintáticas

São ações que buscam corrigir inconsistências que ocorrem devido a erros de digitação e/ou ortografia, ou devido a registros que utilizam diferentes abreviações para

representar a mesma coisa. Um exemplo de erro de digitação é o de uma fonte de dados de clientes conter o nome “Arthur Costa” enquanto em outra existe um cliente com o nome “Artur Costa”. Se os endereços e os números de telefone são idênticos, provavelmente se referem a um único cliente, porém um está errado. Outro exemplo é registros contendo valores como “UFRJ” , “Universidade Federal do Rio de Janeiro” , ou “Univ. Fed. do Rio de Janeiro”.

B) Estruturais

Este tipo de limpeza está relacionado com inconsistências existentes na forma como os dados estão apresentados. Isto inclui:

- Inconsistências de codificação: registros com o mesmo significado podem estar codificados de forma diferente. Por exemplo, uma tabela de clientes pode conter entradas “m” e “f” para representar o sexo masculino e feminino, enquanto outra utiliza “1” e “0”, e uma terceira usa “x” e “y”;
- Inconsistências de tipos de dados: registros de diferentes fontes podem estar representados por conjuntos de caracteres distintos como EBCDIC ou ASCII, ou podem possuir tipos de dados distintos, como uma determinada informação sendo representada como inteiro em uma tabela e como real em outra. Conversões de tipos de dados devem ser efetuadas para converter de um tipo ou formato para o outro;
- Inconsistências de representação de unidades: informações numéricas podem ser registradas com diferentes formatos utilizando diferentes conjuntos de unidades. Por exemplo, uma tabela pode conter valores em real, enquanto outra possui valores em dólar. Outro exemplo é a representação de vendas em milhões em uma determinada fonte de dados, que pode utilizar o valor “9” para indicar “9.000.000”, enquanto em outra, as vendas são representadas em centenas, com o valor “9” representando “900”.

C) Semânticas

Esta categoria de limpeza lida com inconsistências decorrentes de diferentes interpretações de um dado. Isto abrange:

- Inconsistências de definição: registros podem possuir diferentes definições ou significados, embora compartilhem o mesmo nome;
- Inconsistências de integridade referencial: determinados dados em uma tabela podem não encontrar correspondentes em uma outra tabela. Por exemplo, se em uma tabela de fatos VENDAS existe um produto de número 1234567, este produto deve existir na dimensão PRODUTO, caso contrário perde-se a informação do que foi vendido. Assim, se um usuário realizar uma consulta sobre estas tabelas, esta venda será omitida sem sequer o usuário percebê-la;
- Inconsistências devido à reutilização de chaves e a não-unicidade de identificadores: Alguns sistemas legados não realizam operações históricas por mais de um determinado período (p.ex., 90 dias) . Isto possibilita a reutilização de chaves nestes sistemas, o que constitui uma das mais críticas situações de "sujeira" dos dados. Uma outra situação que também ocorre com freqüência é a existência de várias entradas de registro para uma mesma entidade, como exemplo, um cliente com vários códigos. Estas situações já causam prejuízos nos sistemas transacionais, e distorcem de forma significativa as análises em um ambiente de data warehouse;
- Inconsistências devido a valores nulos: Os nulos podem se tornar um grande problema, pois muitos sistemas legados não possuíam forma de representá-los. Para lidar com nulos, os programadores utilizavam valores como 9/9/99 para datas e –1 para, por exemplo, um número de produto. Como todas as interações com os dados passavam pelos programas, era fácil aplicar uma interpretação apropriada para o nulo. No entanto, se agora extrairmos estes dados para um data warehouse sem tratá-los, eles passarão a ser enxergados como valores legítimos. As análises serão efetuadas de maneira errada e o ambiente de data warehouse não atingirá seus objetivos;
- Inconsistências em campos texto: registros podem usar diferentes representações para os dados de um campo texto. Por exemplo, uma tabela pode conter o nome completo de um cliente enquanto em outra só possui o primeiro e o último nome.

Com tantos exemplos de "sujeira" nos dados, somos levados a pensar o porquê de tais dados não terem sido limpos diretamente em suas origens, cortando-se assim o

“mal pela raiz”. No entanto, tal iniciativa não trará grandes benefícios ao sistema legado e pode até ser altamente custosa ou mesmo inviável. Em alguns casos, a limpeza pode e até deve ser efetuada no próprio sistema de origem; porém, concentrar todos os esforços para limpá-los em sua origem não é o caminho.

Desta forma, o melhor momento que se tem para realizar o processo de limpeza dos dados é entre o processo de extração e carga no data warehouse; e o melhor local é no ambiente intermediário utilizado para a construção do ADW.

No que diz respeito ao modo como esta limpeza deve ser realizada, inúmeras regras de transformação dos dados deverão ser discutidas, implementadas e documentadas na forma de metadados. Várias ferramentas de apoio ao processo de conversão de dados para um data warehouse oferecem suporte a estas atividades.

A figura 3.2 ilustra o processo de limpeza dos dados para carga em um ADW.

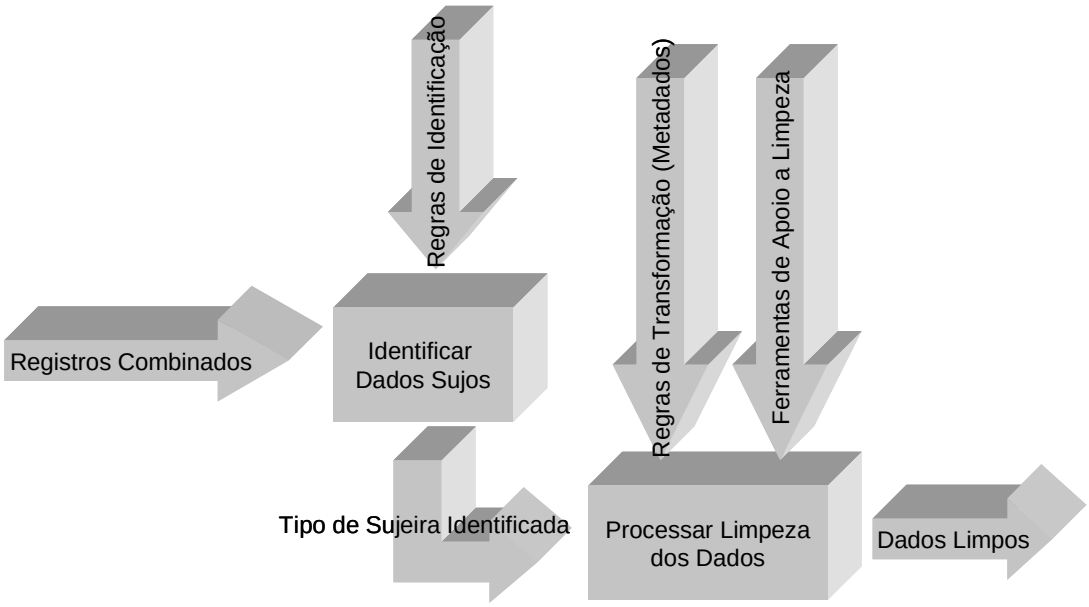

Figura 3.2 - Decomposição do Processo de Limpeza dos Dados

3.1.2.3 Derivação e Sumarização

A derivação consiste nas transformações (p.ex., cálculos) que devem ser realizadas sobre os dados para aplicar as regras de negócio que foram identificadas

durante a fase de requisitos. Como exemplo, poderíamos citar o preenchimento de um atributo novo que representa a margem de lucro de um determinado produto e que deve ser calculado com base nos respectivos atributos de custo e preço de venda deste produto.

A sumarização é a transformação de informações de um nível mais baixo de granularidade para um nível mais alto. Isto evita a redundância e o desperdício de tempo com os cálculos mais comuns. Consiste de duas operações básicas: a agregação e o balanceamento. A agregação é o processo de transformar dados detalhados em entidades que são mais úteis para a análise. O balanceamento tem o propósito de verificação, e basicamente resume os dados detalhados e os compara com um relatório já existente (GOODYEAR *et al*, 1999).

3.1.2.4 Atualizações do histórico

São alterações nos dados do data warehouse decorrentes de atualizações ou identificação de erros. Dois métodos de ajuste são possíveis:

- Atualização manual: o usuário faz todas as mudanças manualmente, primeiro atualizando o nível mais baixo dos dados e depois atualizando os agregados.
- Atualização automática: através do monitoramento dos dados e da leitura dos *logs*, as mudanças são apontadas e todas as atualizações necessárias são realizadas. Embora seja um método extremamente rápido e transparente para o usuário, ele é altamente complexo, pois a implementação de mecanismos de monitoramento dos dados e atualizações automáticas é uma tarefa bastante difícil.

3.1.2.5 Ordenação

É a tarefa de ordenar os registros de dados por um atributo específico. Este passo pode ser realizado através do uso de um *software* de gerenciamento de data warehouse ou através de ferramentas de ordenação especializadas e com alta performance.

3.1.3 Carga

Esta é a última parte do processo de povoamento do ambiente de data

warehouse. Neste momento serão executadas as últimas etapas de preparação dos dados, a carga propriamente dita e algumas atividades posteriores à carga, como o tratamento dos dados rejeitados e o processo de certificação da qualidade dos dados carregados.

As seguintes questões devem ser consideradas em relação à carga de dados (GOODYEAR *et al*, 1999):

- *A janela de tempo disponível para a realização da carga deve ser determinada.* Sendo um ambiente de consultas, não é necessário que o DW permaneça operacional 24 h por dia. No entanto, muitos projetistas - motivados mais pelo estímulo de auto-superação do que propriamente por necessidade de negócio - estão definindo arquiteturas capazes de diminuir (e muito) a janela *off-line* do sistema, podendo até mesmo manter o DW operacional durante todo o dia, sem prejuízo para a carga de dados. De qualquer forma, qualquer que seja a janela de tempo, o ambiente físico deve ser bem projetado e testado para atender a este requisito (ex.: utilização de espelhamento, etc.);
- *A freqüência com que as operações de cargas serão realizadas também deve ser determinada.* Isto depende muito da taxa de crescimento ou mudança das tabelas de dados do ambiente operacional. Por exemplo, algumas tabelas podem precisar ser carregadas somente uma vez por mês, enquanto que outras que sofrem mudanças mais rapidamente precisam ser carregadas diariamente;
- *A performance da carga é altamente dependente da disponibilidade de recursos de sistemas.* Limitações de memória e disco são as causas mais comuns de degradação de performance e falha na carga.

Basicamente, existem dois tipos de gerações de carga:

- *Bulk Load*: carrega o conjunto inteiro de registros no DW. Em geral, mecanismos de banco de dados apropriados são utilizados para realizá-la de forma eficiente;
- *SQL Load:* identifica as mudanças ocorridas desde a última carga e as carrega para dentro do DW.

Podemos decompor o processo de carga de dados da seguinte forma:

1. *Criar imagem de registros*

   Esta etapa deve assegurar que os dados tratados durante o processo de extração e limpeza estão compatíveis com os registros de DW; caso não estejam é o momento de fazê-lo.

2. *Criar agregações e suas respectivas chaves*

   É recomendado que a criação dos agregados seja efetuada antes da carga dos dados, visto que é um processo extremamente custoso, não devendo causar impacto no desempenho do ambiente de data warehouse.

3. *Carregar registros*

   Consiste no processo de carga propriamente dita, onde todos os recursos serão utilizados. Recomenda-se que os índices sejam inibidos nesta etapa por questão de performance. A carga dos dados com integridade referencial é fundamental neste momento, pois será a última oportunidade de identificação de inconsistências. Depois de carregado, o data warehouse possuirá milhões de registros, e encontrar inconsistências será praticamente impossível.

4. *Tratar registros rejeitados*

   Na maioria das vezes, haverá registros rejeitados no processo de carga. Cada um destes registros deverá ser analisado para que a causa da rejeição possa ser identificada e corrigida. Após isto, um novo processo de carga será realizado para processar estes registros. O esforço dispensado nesta atividade será inversamente proporcional à qualidade da realização das etapas de extração e transformação dos dados.

5. *Construir índices*

   Depois dos dados terem sido carregados, os índices devem ser reconstruídos. Atualmente, há sistemas gerenciadores de bancos de dados que possuem a opção de desabilitar os índices em um dado instante. Isto é bastante útil pois, desta forma, não se perde tempo com a reconstrução dos índices das tabelas.

6. *Assegurar a qualidade dos dados*

   Esta etapa tem a finalidade de verificar a qualidade dos dados que foram carregados no data warehouse. Isto pode ser feito manualmente, através da

emissão de relatórios para a checagem de determinados valores, ou através de alguma comparação por amostragem dos valores do data warehouse com os dados oriundos dos sistemas legados.

7. *Divulgar a carga*

   É a formalização da liberação de uma nova versão do data warehouse com os dados atualizados.

3.2 Importância dos Metadados

No desenvolvimento de um data warehouse, a gerência dos metadados constitui um fator crítico de sucesso para uma implantação consistente. Além de serem essenciais no processo de transformação dos dados em informações, eles permitem que o usuário utilize o data warehouse de forma pró-ativa a fim de maximizar o resultado das análises. Além disso, são fundamentais no processo de prover subsídios para definição, construção, gerência e manutenção de um data warehouse (MEYER, CANNON, 1998).

Sem os metadados, o data warehouse pode ser considerado apenas como um simples repositório de dados, onde o poder de análise por parte do usuário é extremamente limitado. Os usuários necessitam conhecer a estrutura e o significado dos dados para que consigam atingir os resultados esperados da forma mais lógica e intuitiva possível.

Para melhor compreender o papel dos metadados dentro de uma arquitetura de data warehouse, definimos dois tipos: metadados de negócio e metadados do data warehouse propriamente ditos (PLATINUM, 1999).

Os metadados de negócio visam à apresentação dos conceitos dos objetos de informação ao usuário final da forma mais adequada e familiar possível. Além disso, eles devem fornecer subsídios para uma análise apropriada dos dados; como exemplos, a forma como um dado derivado foi calculado, a data/hora que um relatório foi criado e quão atualizados estão os dados contidos no data warehouse. Um papel importante deste tipo de metadado é auxiliar o usuário a compreender as mudanças que os dados sofrem ao longo do tempo, que podem ser fruto de redefinições de conceito ou mesmo do negócio da empresa.

Os metadados do data warehouse são os que interessam no escopo de nosso

trabalho. Eles abrangem todos os aspectos técnicos e funcionais do data warehouse, constituindo uma visão abstrata do mesmo, definindo todos os elementos e a forma como interagem entre si. Tudo dentro de um data warehouse, com exceção dos dados, pode e deve ser considerado metadado.

Iremos focar nos metadados presentes nos processos de transformação e carga dos dados no data warehouse (BERSON, SMITH, 1997) :

- Durante o processo de leitura e extração dos dados dos sistemas legados, é necessário ter acesso aos metadados dos mesmos, como os esquemas das aplicações, seus dicionários de dados, as regras de negócio, restrições de integridade, freqüência das atualizações, e quaisquer outros documentos que auxiliem no conhecimento destes sistemas;
- O próprio processo de extração abrange alguns tipos de metadados, como quando e como será realizada a captura dos dados de cada sistema fonte, qual a plataforma em que se encontram, e algumas informações específicas sobre esta plataforma;
- Em relação ao repositório do data warehouse, constituem metadados o novo esquema de dados e seu dicionário de dados, definições de visões e *stored-procedures*, informações sobre particionamento e índices, etc.;
- A etapa de transformação dos dados é bastante rica em metadados, pois ela apresenta as transformações pelas quais os dados passam para se tornarem informações de relevância para o usuário. Deve-se mapear cada dado presente no data warehouse, até o seu sistema de origem, informando de que sistemas(s) se originou, de que tabela(s), de que atributo(s), etc., documentando todas as transformações que sofre este dado (conversão de formato, derivação, limpeza, etc.);
- Também devem ser mantidas informações sobre agregados, freqüência de atualização dos mesmos e estatísticas de uso por parte do usuário final;
- Dados sobre desempenho e monitoramento do ambiente de data warehouse também constituem importantes metadados, pois através deles poderão ser efetuados ajustes nos processo de extração, carga e uso do data warehouse;
- Dados que identificam questões relativas a qualidade dos dados, detectadas durante o processo de povoamento do data warehouse também devem estar disponíveis para os usuários, para que estes possam julgar a consistência de suas análises.

CAPÍTULO 4

PROJETO FÍSICO DE UM DATA WAREHOUSE

Antes que as etapas de povoamento inicial do ADW sejam realizadas, é necessário que se elabore o projeto físico deste ambiente de modo que seja capaz de suportá-las. Desta forma, recomenda-se que, após a criação do esquema dimensional do data warehouse, se inicie a etapa de conversão do mesmo para o esquema físico. Este deve refletir ao máximo o esquema lógico, buscando uma correspondência um para um de tabelas de dimensões e de fatos. Porém, algumas mudanças nas estruturas das tabelas e colunas são necessárias para acomodar as idiossincrasias do sistema gerenciador de banco de dados e das ferramentas de acesso escolhidas. Isto refletirá diretamente na estratégia de criação de índices, na integridade referencial e no plano de particionamento de tabelas.

São estes detalhes de especificação de características físicas que constituem a principal diferença entre o esquema lógico e o físico de um data warehouse, e que serão abordados por este capítulo. Também será discutida a estratégia de criação de agregados para se obter uma melhor performance do data warehouse, e ainda, uma breve descrição da forma como toda a infra-estrutura deve ser planejada para que não imponha limitações à evolução inicial do ambiente de data warehouse.

4.1 Definição do Esquema Físico de um Data Warehouse

Um projeto físico bem implementado de um data warehouse pode determinar a diferença entre o sucesso e o fracasso do mesmo. Como a implementação do DW é altamente dependente de componentes individuais do projeto (KIMBALL *et al*, 1998) - como o modelo lógico de dados, o sistema gerenciador de banco de dados utilizado, volume de dados, padrões de uso e ferramentas de acesso – é fundamental que o projeto físico leve em consideração estes fatores desde o início, para que não ocorram surpresas no decorrer da mesma.

Como já foi dito, o esquema físico de dados deve ser o mais simples e próximo possível do esquema lógico. No entanto, pequenas diferenças são inevitáveis, pois são

necessárias para que suportem requisitos específicos das ferramentas utilizadas, para otimizar a performance das consultas realizadas, e também para fazer com que o ciclo de manutenção do data warehouse possa ocorrer dentro de uma janela de tempo aceitável.

A seguir, descreveremos as principais etapas do processo de transformação do esquema lógico no esquema físico de dados (KIMBALL *et al*, 1998):

4.1.1. Definição de Padrões

Em qualquer projeto de implementação de banco de dados, busca-se a consistência entre os objetos de dados que o compõem. Em um projeto do porte de um data warehouse, isto não é diferente. Dois importantes grupos de padrões devem ser definidos: aqueles relativos à nomenclatura de objetos de dados e os referentes ao nome dos arquivos físicos e suas localidades.

Deve-se buscar que os nomes do esquema físico de dados reflitam os nomes do esquema lógico, e sejam o mais descritivos possíveis. O uso de uma ferramenta de modelagem auxilia bastante neste processo.

Um exemplo da importância da padronização da nomenclatura é o que permite diferenciar as tabelas auxiliares dos processos de transformação das tabelas do data warehouse propriamente ditas.

4.1.2 Criação de Chaves

Nesta etapa são criadas as chaves primárias e estrangeiras especificadas no modelo lógico de dados. A integridade referencial é extremamente importante para um data warehouse, visto que este é um ambiente de consultas intensivas, onde inúmeras ligações (“joins”) entre tabelas são realizadas. Desta forma, a tabela de fatos possuirá uma chave composta pelas chaves das tabelas dimensionais a ela relacionadas, tornando muito mais eficiente a recuperação de dados destas tabelas.

Também serão criadas novas chaves identificadoras das dimensões do ADW. A nova identificação, em geral, é decorrente dos seguintes fatores (INMON, 1997):

- necessidade de remapear a chave para evitar a dependência da chave original. Isto ocorre quando existe a possibilidade de alteração de chave e é necessário evitar a sua reutilização. O reuso de chaves é comum no ambiente operacional

devido a sua pequena periodicidade de armazenamento. O ADW, ao contrário, armazena os registros por um longo período, exigindo uma nova chave para evitar duplicidades e inconsistências para as consultas (KIMBALL, 1998);

- remapeamento de chaves, reduzindo chaves longas para obter um melhor desempenho nas consultas;
- estabelecimento de chaves genéricas, permitindo mudanças na descrição dos itens sem provocar alterações nas chaves. A solução normalmente adotada no nível físico, é acrescentar dois ou mais dígitos ao final da chave original. Estes novos dígitos indicam a versão do item. As chaves genéricas permitem o rastreamento de modificações pela generalização da chave primária (KIMBALL, 1998).

É essencial que os tipos de dados escolhidos para as colunas que são chaves sejam os que trarão maior benefício de performance no momento da realização de consultas. Em geral, o tipo de dados mais indicado é o inteiro (“integer”). Porém, há casos onde o tipo de dados caracter (“char”) de tamanho fixo pode ser mais eficiente. Uma análise das características do SGBD utilizado deve ser realizada para que se opte pela melhor solução.

Também em busca de um melhor desempenho, deve ser avaliada a substituição das chaves compostas de colunas do tipo DATA por chaves artificiais (“surrogate keys”) compostas de colunas do tipo inteiro. Como grande parte das consultas em um ADW envolvem colunas do tipo DATA em suas cláusulas, a ligação (“join”) com a tabela dimensional de tempo fica muito mais eficiente.

4.1.3 Criação de Mecanismo de Controle dos Processos de Transformação e Carga

Para que seja possível controlar a execução dos processos de transformação e de carga do ADW é necessário que se crie uma tabela de controle para este fim. Desta forma, esta tabela conteria atributos como: identificador do processo, datas de início e fim de sua execução, estado do processo (finalizou com sucesso ou não), data de atualização do último registro processado, etc. É bastante útil também que para cada tabela do ambiente sejam adicionados mais dois atributos que permitirão um maior

controle da inserção/atualização de seus registros – data de criação e data de atualização.

4.1.4 Indexação

Em um ambiente de consultas como um ADW, a questão de performance é uma preocupação constante. Um dos principais métodos para tratar esta questão é o de criação de índices. Uma estratégia de indexação bem planejada irá minimizar o número de índices e irá otimizar todos os acessos mais críticos aos dados. É tentador que se procure adicionar índices que satisfaçam qualquer tipo de acesso a uma tabela. Porém, deve ser claro que quanto mais índices uma tabela tiver, pior será o tempo de carga da mesma, visto que os índices necessitam ser reestruturados quando ocorrem inserções e exclusões de registros (GOODYEAR *et al*, 1999). Mais índices também representam mais custos de administração do ambiente, pois cada índice deve ser considerado no momento de estimar o tamanho das estruturas de armazenamento, e de planejar a acomodação do crescimento do ADW.

Recomendamos as seguintes ações para atingir uma melhor estratégia de criação de índices (GOODYEAR *et al*, 1999):

- em geral, quando o data warehouse é implementado sobre um sistema gerenciador de banco de dados relacional, índices são criados nas colunas definidas como chaves primárias das tabelas. Índices adicionais para as tabelas de fatos e dimensões serão necessários dependendo da forma esperada de acesso aos dados;
- os índices criados para as chaves primárias se baseiam na ordem na qual as colunas foram declaradas. Como a maioria das consultas em um ADW contém restrições nos campos de datas, é recomendado que as colunas de datas sejam as primeiras da chave primária. Isto também aumenta a performance dos processos de carga, visto que as cargas incrementais, em geral, são baseadas em datas;
- use poucos índices, ou até mesmo nenhum, para tabelas com uma alta taxa de inserções/exclusões, para acessos de grande volume em tabelas em uma ordem específica, e para tabelas dimensionais pequenas, mas altamente acessadas. Nestes casos, o acesso sequencial pode ser aceitável;
- a forma como os usuários acessarão os dados não pode ser a priori precisamente definida. Mas, não é por esta razão, que índices devam ser criados para cada

coluna das tabelas do data warehouse, em antecipação a forma como serão acessadas. Isto não é viável;

- deve-se estar atento ao fato que os sistemas gerenciadores de banco de dados tradicionais só permitiam a utilização de um único índice em cada passo de uma consulta. Porém, as últimas versões de grande parte destes sistemas contêm otimizadores de consultas que fazem uso de múltiplos índices;
- escolha o tipo de índice adequado para os acessos mais críticos (KIMBALL *et al*, 1998):
    - Índice B-Tree : é bastante eficaz para colunas de alta cardinalidade. É o tipo de índice padrão para a maioria dos sistemas gerenciadores de bancos de dados relacionais;
    - Índice Bitmap: ao contrário dos índices B-Tree, este índice é mais apropriado para colunas com baixa cardinalidade. Basicamente, constitui uma sequência de bits para cada valor possível de uma determinada coluna. A grande vantagem deste índice é a economia de espaço que proporciona para colunas com baixa cardinalidade. Ele permite ser processado com pouco ou nenhum I/O, além de podermos efetuar operações de lógica booleana em sua própria estrutura. Este índice é recomendado para colunas de tabelas dimensionais que servem de filtro para consultas;
    - Índice Hash: pode ser utilizado em casos extremos de alta cardinalidade. Ao invés de construir um índice, ele utiliza uma fórmula matemática para calcular exatamente em qual página o registro está localizado. É suportado por poucos sistemas gerenciadores de bancos de dados;
    - Índice Join: este tipo de índice tem o intuito de otimizar as consultas que envolvam diversas tabelas, como é o caso do data warehouse com o modelo estrela. Ele é estruturado sobre condições de ligações (“joins”) entre duas ou mais tabelas do ADW. Estas condições são colocadas em conjunto no índice, onde cada referência nada mais é do que um identificador de tuplas de ligações. Como é um índice composto por mais de uma tabela, ele é complexo e mais difícil de se criar e manter que um índice comum, de uma única tabela;

- Outros Tipos de Índices: alguns sistemas gerenciadores de bancos de dados utilizam estruturas de índices proprietárias. Estas estruturas devem ser um dos principais fatores de análise quando da seleção do sistema a ser utilizado.

É extremamente importante que a estratégia de indexação utilizada seja revista regularmente, com base nas formas de acesso aos dados, para que índices que não estão sendo utilizados sejam excluídos e novos índices sejam criados (KIMBALL *et al*, 1998).

Outro ponto importante a ser destacado é a existência de índices no momento da carga do ADW, pois o custo de reestruturação dos mesmos pode ser muito alto e degradar bastante a performance destes processos. Segundo os principais fornecedores de SGBDs, em tabelas onde a carga de novos dados representa mais de 10% do total de registros da mesma, é mais eficiente excluir os índices antes das inserções, e recriá-los após este processo. Atualmente, já existem alguns SGBDs que permitem desabilitar os índices no momento de carga de uma tabela.

4.1.5 Dimensionamento do Banco de Dados

Para calcular a quantidade de investimentos em *hardware* que serão necessários para suportar o ambiente de data warehouse é preciso que se estimem os requisitos de armazenamento do mesmo. É importante que estas estimativas estejam bem embasadas para que não se subestimem ou superestimem os recursos necessários.

As seguintes ações devem ser tomadas para uma estimativa inicial (KIMBALL *et al*, 1998):

- Estimar o tamanho das linhas, considerando que colunas do tipo VARCHAR ou que aceitem nulos podem ser menores do que o tamanho máximo das mesmas. Este percentual pode ser calculado com base na análise de uma pequena amostra;
- Para cada tabela, estimar o tamanho da mesma com a carga inicial completa (dados históricos) e como ela irá evoluir com as cargas incrementais. É importante que aqui seja considerado o expurgo de dados após um determinado período de armazenamento;

- Para os SGBDs tradicionais, reserve o mesmo espaço para o armazenamento dos índices que o ocupado pelos dados das tabelas em que se baseiam;
- Para áreas temporárias que serão utilizadas para ordenação, geralmente é necessário o dobro de espaço ocupado por um índice para construí-lo. Para operações de ordenação/grupamento de dados de grandes tabelas, a área temporária deve ser dimensionada para que, no mínimo, tenha o tamanho destas tabelas;
- Reservar espaço para as tabelas de metadados;
- As tabelas de agregados - as quais discutiremos mais adiante neste capítulo – tendem a ocupar um espaço considerável, e dependem do grau de agregação dos dados armazenados, a esparsidade destes dados e da profundidade das hierarquias. Em geral, podemos considerar, assim como os índices, que ocuparão o mesmo espaço de armazenamento das tabelas em que se baseiam.

É importante notar que os tamanhos das tabelas dimensionais são desprezíveis se comparados com as tabelas de fatos. Após as tabelas de fatos, o maior motivo de preocupação em relação a espaço é relativo ao tamanho a ser ocupado pelos índices construídos sobre estas tabelas.

4.1.6 Particionamento

Desde que o SGBD utilizado suporte, uma estratégia de particionamento apropriada pode incrementar bastante a performance do ADW. Em geral, somente tabelas de fatos, seus índices e grandes tabelas dimensionais são candidatos ao particionamento (KIMBALL *et al*, 1998). Uma tabela particionada se parece com uma tabela comum, porém é gerenciada em múltiplos blocos de dados. As vantagens deste mecanismo são basicamente duas :

a) Uma consulta irá acessar somente as partições necessárias para a sua resolução.

Se o modo de particionamento for o mais adequado, isto ocasiona grandes benefícios de performance. Em geral, a melhor maneira de se particionar uma tabela é através da segmentação por tempo (mês, bimestre, ano, etc.). Porém, nada impede que as partições possam ser definidas por regiões, linha de produtos, etc. Para se obter bons resultados com o particionamento, basta definir um critério que determine de forma única a qual partição um determinado registro irá pertencer.

b) A gerência de uma partição é muito mais eficaz que a gerência de toda a tabela.
Isto reduz bastante os custos de manutenção nos casos em que somente uma porção de dados mais recentes estiver ativa e acessível. Além disso, cargas e backups são mais facilmente realizados visto que tratam com porções menores de dados.

No entanto, é preciso que a estratégia de particionamento seja bem planejada, para que não acarrete em algumas desvantagens. Se o ADW contiver um número significativo de consultas que precisem acessar diversas partições de uma única vez, a performance pode ser degradada, pois é bem mais rápido acessar uma tabela não particionada (BERSON, SMITH, 1997).

Em configurações de arquitetura onde o paralelismo é implementado fisicamente, o particionamento é um requisito básico para um melhor proveito das mesmas.

4.1.7 Acompanhamento do Uso do ADW

O ambiente de um data warehouse se encontra sempre em constante evolução. Nenhuma tarefa ao longo do ciclo de vida de um ADW pode se encarada como finalizada. Na medida em que o ADW evolui, cada uma destas tarefas deve ser revista e adequada às necessidades do momento. Com o projeto físico de um ADW não é diferente.

Informações como padrões de uso, tempos de resposta, concorrência de usuários, utilização de índices, tempos de carga, entre outras, são de extrema importância para que o administrador do data warehouse possa configurar o ambiente de modo a lhe tirar o máximo de proveito. Além disso, elas são necessárias para que ele possa aferir qual a taxa de crescimento do ADW a fim de estimar que investimentos serão necessários para suportar sua evolução (KIMBALL *et al*, 1998).

4.2 Agregados

Sem dúvida alguma, o recurso mais eficaz para otimizar a performance de um ambiente de data warehouse é a utilização de agregados (KIMBALL *et al*, 1998). Isto se

deve ao fato da maior parte das consultas realizadas em um ADW analisarem somente um subconjunto dos dados em um nível menor de detalhe, ou seja, um nível onde os registros já se encontram agrupados de uma determinada maneira. Como, em geral, as tabelas de um data warehouse contêm um volume de dados muito alto, e este tipo de consulta tende a ser muito complexo em termos de estrutura e cálculo, a pré-computação destes registros agregados provoca um efeito bastante positivo em termos de desempenho do ambiente; reduz consideravelmente o número de registros resultantes de uma consulta e também o tempo de processamento (pode-se obter uma redução de dez a mil vezes do tempo de processamento, através do uso dos agregados apropriados).

Para obter sucesso, um plano de agregação deve, além de buscar um melhoria de performance significativa no acesso aos dados armazenados, atingir as seguintes metas (KIMBALL *et al*, 1998):

- possibilitar ganhos substanciais de performance para o maior número possível de consultas realizadas;
- causar o menor impacto possível nos processos de extração de dados. Inevitavelmente, grande parte dos agregados precisam ser criados ou atualizados sempre que novos dados são carregados nas tabelas; porém, a especificação destes processos deve procurar ser o mais automática possível. Esta é uma tarefa bastante árdua em data warehouses grandes, onde normalmente a janela de tempo para a transformação dos dados é muito restrita;
- balancear a questão de aumento de performance e aumento de espaço de armazenamento, evitando adicionar registros agregados que não trarão um retorno adequado em termos de melhoria no tempo de acesso aos dados. De uma forma geral, admite-se que os agregados ocupem, no máximo, o mesmo espaço de armazenamento ocupado pelas tabelas em que se baseiam. Cabe ressaltar que, ambos os fatores (performance e espaço de armazenamento) podem determinar o sucesso ou o fracasso de um projeto de data warehouse. Não há outra forma mais eficiente de se obter os benefícios de performance senão pelo uso de agregados. Por outro lado, um plano mal feito pode requerer um aumento significativo de espaço em disco, elevando tremendamente o custo do data warehouse;
- ser completamente transparente para os usuários e para os desenvolvedores das aplicações, a não ser pela percepção dos benefícios de performance alcançados. Em

termos práticos, nenhum SQL de usuário ou aplicação deve referenciar diretamente os agregados. Esta é uma questão que só pode ser tratada caso o ADW possua um mecanismo de navegação de agregados, como discutiremos mais adiante. Porém, em nenhuma hipótese, esta transparência pode se tornar impeditiva para a criação de agregados no ADW;

- reduzir ao máximo o impacto nas tarefas sob responsabilidade dos administradores do ambiente de data warehouse.

Além do aumento de performance, os agregados também produzem um outro benefício: a possibilidade de garantir que eles englobam conceitos corretos. Por exemplo, a definição de *clientes ativos* pode ser óbvia para os gerentes de relacionamento de uma empresa, mas é arriscado permitir que cada usuário crie este grupo sempre que uma análise é realizada.

Nos itens que se seguem, trataremos das principais questões associadas com a criação de agregados em um ambiente de data warehouse.

4.2.1 Escolha dos Agregados a Serem Criados

Como já foi dito, a estratégia de criação de agregados deve ser bastante criteriosa para que o seu custo/benefício seja compensador. Dois aspectos devem ser considerados quando da seleção das agregações a serem construídas (KIMBALL *et al*, 1998):

- as questões mais freqüentes de negócio;
- a distribuição estatística dos dados armazenados no data warehouse.

Cada dimensão deve ser cuidadosamente analisada para determinar quais são os atributos mais utilizados em agrupamentos de dados. Feito isto, deve-se escolher quais serão utilizados em conjunto. É muito importante que esta análise seja bem feita, pois na medida em que o número de dimensões aumenta, o número de candidatos a agregados aumenta exponencialmente. Por exemplo, para 4 dimensões, cada uma possuindo 3 atributos candidatos à agregação, poderíamos construir até 256 agregados (considerando todas as combinações possíveis entre eles - 4^4). Se aumentarmos o número de dimensões para 8, chegaríamos a 65.536 possíveis agregados (4^8). Certamente, apenas

uma pequena parcela das possíveis combinações destes atributos candidatos será realmente utilizada.

Outro ponto que deve ser considerado é o domínio de valores de um determinado atributo candidato à agregação. Suponha que temos uma tabela que, no maior nível de detalhe, possa possuir até 1.000.000 valores para um determinado atributo. Agora imagine que, agrupando os dados em um nível menor de detalhe, chegamos a quantidade de 500.000 registros diferentes. Mesmo que este nível de agrupamento seja bastante solicitado em consultas, não valeria a pena pré-agregá-lo pois ele não provocaria um aumento de performance significativo. Entretanto, se o número de registros agregados chegasse a 50.000, este mesmo atributo seria um sério candidato à pré-agregação.

Cabe ressaltar que o processo de construção/manutenção de agregados em um ambiente de data warehouse é extremamente dinâmico, devendo ser encarado como uma forma de realizar ajustes de performance no ambiente; ou seja, se as agregações criadas não resultaram em ganhos de performance significativos, deve ser realizada uma revisão do plano de agregações.

4.2.2 Seleção da Técnica de Armazenamento de Agregados a Ser Utilizada

Uma vez escolhidos quais os agregados que serão construídos, deve-se decidir a forma como eles serão armazenados. Existem duas técnicas: inserção de novas tabelas de fatos e de dimensões, ou o uso de colunas de nível para identificar os registros de agregados (Kimball, 1998).

A) Criação de Novas Tabelas de Fatos e de Dimensões

Esta técnica consiste na criação de uma nova tabela de fatos para cada agregação a ser criada, o que na verdade, não passa de uma nova tabela de fatos derivada da tabela de fatos de nível básico. Da mesma forma, a tabela de fatos derivada deve estar associada a uma ou mais tabelas dimensionais derivadas. No entanto, as novas tabelas dimensionais correspondem a versões reduzidas das tabelas de dimensão originais, visto que boa parte dos atributos não fazem sentido para o nível de agregação determinado. Por exemplo, se construíssemos uma agregação sobre uma tabela de vendas de um supermercado, no nível de categoria de produtos, muitos dos atributos da tabela de fatos

deveriam ser suprimidos, pois só fazem sentido quando descrevem produtos individuais.

Cada nova tabela de fatos agregados é totalmente preenchida por registros que não existem nas tabelas de nível básico, pois cada novo registro representa fatos numéricos totalizados pela agregação. As totalizações ocorrem pelo somatório de valores quando o fato for aditivo e por alguma forma particular de processamento quando o fato for semi-aditivo. Quando não puderem ser somados, os valores são de algum modo processados para se definir o valor agregado.

Embora acarrete em um número elevado de novas tabelas, esta técnica é a mais recomendada para projetos de data warehouse devido a uma série de razões:

- apesar do número de novas tabelas crescer bastante, a administração dos agregados é realizada de forma mais modular, pois cada um deles corresponde a tabelas distintas. Desta forma, eles podem ser criados, carregados, eliminados e indexados independentemente;
- elimina o risco do usuário final realizar a contagem dupla de registros;
- as tabelas de agregados podem ser totalmente transparentes aos usuários finais, caso se utilize uma solução de navegação de agregados (que discutiremos mais adiante);
- como o tamanho de campos numéricos na tabela de agregados geralmente é maior que na tabela de fatos de nível básico - pois são somatórios dos fatos individuais – a separação de agregados em tabelas próprias evita que se aumente o tamanho destes atributos na tabela de nível básico só para comportar alguns poucos registros agregados.

Para cada nova tabela de agregados, devem ser criadas chaves artificiais, diferentes das chaves contidas nas tabelas de nível básico.

B) Inserção da coluna *Nível* para registros agregados

Nesta técnica, colunas *Nível* são adicionadas a cada uma das tabelas dimensionais correspondentes à agregação, permitindo que os registros de fatos agregados possam ser armazenados na tabela de fatos original. Em relação à técnica anterior, é criado o mesmo número de registros e, da mesma forma, devem ser geradas chaves para as tabelas de fatos e de dimensões.

A nova coluna descreve o nível de agregação de cada registro na tabela de dimensão. Todos os registros de nível básico contêm o valor indicativo deste nível. Os registros agregados precisam de chaves compatíveis e que não conflitem com as chaves originais de nível básico na tabela de dimensão.

Como os registros agregados estão na mesma tabela dos registros básicos, somente poucos atributos farão sentido para eles. Como estes atributos não podem ser descartados, eles conterão valores nulos ou indicativos de não aplicabilidade (p.ex.: ‘NA’), desperdiçando espaço de armazenamento sem informação útil.

Um outro problema de se compartilhar as tabelas é a possibilidade de se realizar contagem dupla, caso se deixe de restringir o campo Nível a um valor indicativo do nível de hierarquia que se deseja acessar.

4.2.3 Processos de Criação de Agregados

O único requisito básico para a criação de agregados é que eles sejam criados a partir dos dados de nível básico. Sendo assim, a seguir estão descritas variações do processo de agregação (KIMBALL *et al*, 1998):

- as agregações são construídas fora do SGBD destino, através da ordenação dos registros e segmentação dos mesmos com base nas classes de agregação existentes (*break rows*). Como já existem antes da carga do SGBD, podem ser carregados separadamente;
- os agregados são construídos aos poucos, através da consolidação incremental, à medida que os registros básicos vão sendo carregados. Um exemplo desta forma de criação é a construção de registros agregados baseados em meses, enquanto que as cargas no ADW são realizadas diariamente (após a última carga do mês, a agregação está completa);
- construção automática dos agregados como parte integrante do processo de carga. Este método pode ser encarado como uma variação dos dois anteriores, porém sem a existência física dos agregados fora do SGBD destino;
- os agregados são construídos dentro do ADW, usando SQL após a carga dos registros básicos.

A última alternativa é a menos recomendada, pois os passos de criação de um agregado são essencialmente sequenciais, e não relacionais, tornando o processamento dentro do SGBD ineficaz.

A segunda alternativa requer alguns cuidados: conforme o exemplo citado, os registros carregados a cada dia causam inserções e/ou atualizações no agregado que está sendo construído. Se um registro com a mesma chave já estiver presente, então será realizada uma atualização dos fatos desse registro, adicionando os novos valores aos correspondentes já armazenados. Porém, se o registro com a mesma chave não existir, será então inserido o novo registro.

Para manter a flexibilidade e independência necessárias, as chaves geradas para os registros agregados não devem ter relação direta com nenhum atributo de produção.

Um outro ponto que deve ser bastante destacado é o fato de se manter os agregados sincronizados com os dados básicos contidos no data warehouse a cada instante. Caso os dados básicos sejam atualizados e disponibilizados, porém por alguma razão, os agregados não puderem ser, então os antigos agregados devem ser eliminados até que os novos sejam criados e colocados em uso. Senão, eles não refletirão os dados da forma correta. É preferível obter uma degradação na performance do ambiente devido à ausência de um conjunto de agregados a ter que utilizá-los com informações inconsistentes.

4.2.4 Administração de Agregados

Agregações são um recurso dinâmico do data warehouse (KIMBALL *et al*, 1998). De forma similar aos índices, os administradores, com base em estatísticas, podem decidir eliminar um determinado agregado se ele não estiver sendo usado. Afinal, ele pode demandar um grande espaço em disco, além de criar um *overhead* no processo de extração e carga dos dados. Estatísticas também fornecem subsídios para a criação de novos agregados, permitindo que estimativas de tamanho e tempo de construção/atualização sejam efetuadas. Além disso, pode-se estimar a redução no tempo de resposta a determinadas consultas devido à utilização de agregados.

O uso da estratégia de armazenamento de cada agregado em tabelas distintas facilita a manutenção dos mesmos, visto que o administrador de banco de dados do ADW pode criar, alterar e excluir agregados de forma transparente para os usuários.

De acordo com Kimball (KIMBALL, 1998), é razoável planejar, como regra geral, um acréscimo de 100% no espaço em disco para o armazenamento das agregações. Isto significa que, em média, a soma de todas as tabelas de agregados deve ocupar um espaço de armazenamento menor, ou no máximo igual, ao espaço ocupado por todas as tabelas de nível básico. Caso a soma das tabelas agregadas ocupem 25% ou menos em relação às tabelas básicas, é provável que os usuários estejam sendo penalizados com baixas performances nas respostas às suas consultas. Por outro lado, se o total ocupado pelas tabelas agregadas for muito superior ao total ocupado pelas tabelas básicas, isto pode representar um indício que há um excesso de agregados no ambiente e, certamente, boa parte deles não estão sendo utilizados e, outros, apesar de estarem sendo usados, não estão trazendo benefícios de performance para justificar sua existência.

A seguir, estão listados os requisitos sugeridos por Kimball (KIMBALL, 1996-b), para maximizar os benefícios da construção dos agregados, sem que seja necessária uma complexa estrutura de metadados para suportá-los:

- Requisito 1: Os agregados devem ocupar tabelas próprias, separadas dos dados de nível básico;
- Requisito 2: As tabelas de dimensões agregadas devem ser versões reduzidas das tabelas de dimensões básicas correspondentes, contendo apenas atributos que façam sentido no nível de agregação determinado;
- Requisito 3: Deve existir um metadado (o único necessário para esta arquitetura de agregados) que associe a tabela de fatos básica com todas as suas tabelas de agregados derivadas;
- Requisito 4: As tabelas de fatos e de dimensões agregadas devem ser completamente transparentes aos usuários finais. Sendo assim, todos os comandos SQL emitidos pelo usuário devem referenciar somente às tabelas de fatos e de dimensões de nível básico.

Se o ambiente de data warehouse atender a estes requisitos, ele está apto a ter um navegador de agregados que permita a utilização otimizada dos agregados existentes, de modo totalmente transparente para o usuário.

4.2.5 Navegador de Agregados

O navegador de agregados é uma camada de software localizada entre o usuário final e o SGBD, de forma que permita que o usuário final possa usufruir dos benefícios dos agregados, porém sem que se preocupe com a existência dos mesmos (KIMBALL, 1996-a).

De uma forma resumida, o navegador de agregados trabalha da seguinte maneira: com base em estatísticas de consulta e em metadados, ele intercepta o comando SQL originado pelo usuário final, e o altera de forma a utilizar o melhor agregado disponível para este tipo de consulta. Na verdade, basta substituir os nomes das tabelas básicas pelos nomes das tabelas agregadas; os nomes dos atributos são mantidos, pois são correspondentes às tabelas básicas.

Do ponto de vista lógico, o mais natural é que o navegador de agregados seja parte integrante do SGBD, visto que a manutenção das metatabelas essenciais é de responsabilidade dos DBAs, além do fato que devem estar centralizadas em um único servidor.

4.3 Infra-estrutura

No projeto de implementação de um ambiente de data warehouse, devem ser consideradas várias questões relativas à arquitetura técnica do ambiente, tais como:

- a plataforma de *hardware* que será utilizada;
- o sistema gerenciador de banco de dados que irá suportar o data warehouse;
- ferramentas de acesso do usuário;
- o *hardware* e o *software* que suportarão o repositório de metadados;
- os sistemas de gerenciamento que permitem realizar a administração de todo o ambiente.

Segundo Kimball (KIMBALL *et al*, 1998), o princípio básico a ser considerado quando da seleção da infra-estrutura técnica é o fato de que o data warehouse crescerá rapidamente nos primeiros 18 meses, em termos de dados e de uso. Esta observação não

deve ser nunca subestimada, para que em nenhum momento a arquitetura técnica planejada imponha limitações à evolução do data warehouse.

Mesmo em relação à parte mais técnica do desenvolvimento do ADW, os requisitos do negócio continuam a ser o fator determinante para as escolhas a serem feitas; são eles que determinam o nível de detalhe dos dados e o tempo de retenção dos mesmos. Além disso, são eles que determinam o grau de complexidade das regras de transformação e a periodicidade de carga dos dados. Todos estes fatores influenciam diretamente na escolha do *hardware* e dos dispositivos de armazenamento. A seguir, listamos os principais fatores que influenciam na seleção da plataforma técnica (KIMBALL *et al*, 1998):

- **"Tamanho" dos Dados:** a quantidade de dados que se necessita armazenar é determinada pelas questões de negócio que o data warehouse se propõe a resolver. No dimensionamento da capacidade de armazenamento do ADW, deve ser considerada a taxa de crescimento do mesmo, e ainda, a criação de índices e agregados, que em nenhum momento podem ser subestimados;
- **Volatilidade:** indica o grau de dinamismo das bases de dados, ou seja, a freqüência das inserções/atualizações, além da duração da janela de carga;
- **Número de Usuários:** embora pareça óbvio, é importante que seja considerado o número de usuários que poderão acessar o ambiente ao mesmo tempo, além dos picos de atividade. Um número alto de usuários realizando consultas complexas concorrentemente, tende a degradar a performance do ambiente de consulta muito rapidamente, no caso de este não ter sido bem dimensionado;
- **Número de Processos de Negócio:** este fator impacta diretamente a complexidade do ambiente, pois determina o número de processos de transformação e carga, a janela de carga demandada, etc.

Além destes fatores, há ainda questões como disponibilidade de suporte técnico na região, inter-dependência da plataforma de *hardware* com os *softwares* selecionados e ainda, talvez o mais importante fator, a disponibilidade de recursos financeiros a serem dispendidos no projeto.

4.3.1 Plataforma de Hardware

O ponto-chave para a seleção do servidor que comportará o data warehouse é o fato dele suportar, de forma eficiente, o grande volume de dados e a quantidade de consultas complexas que serão executadas no ambiente. Além disso, ele deve possuir escalabilidade para acomodar o crescimento contínuo do data warehouse. O equilíbrio entre o poder de processamento da plataforma e a largura de banda de entrada/saída também deve ser atingido para que se consiga obter uma melhor performance do ambiente.

Outro ponto a ser considerado é a disponibilidade de ferramentas e soluções que existem no mercado compatíveis com o *hardware* a ser escolhido. Questões como paralelismo de *hardware* e dispositivos de armazenamento (taxas de transferência, frequência de acesso, configuração de discos para garantir redundância e disponibilidade) também devem ser analisadas com atenção, no momento da escolha da plataforma de *hardware* a ser utilizada.

4.3.2 Plataforma de Dados

Muitos dos fatores que influenciam a decisão da escolha da plataforma de *hardware* também se aplica à seleção do sistema gerenciador de banco de dados a ser utilizado. Porém, neste caso, talvez a maior consideração seja a escolha entre o ambiente relacional e o ambiente multidimensional.

Enquanto que a tecnologia relacional é mais adequada para grandes volumes de dados, o ambiente multidimensional favorece a performance de consultas do usuário final sobre dados contidos em um modelo dimensional, onde os fatos estão pré-armazenados com todas as combinações válidas para as dimensões. Análises mais complexas são realizadas mais facilmente em um ambiente multidimensional. Por outro lado, este ambiente possui limitações quanto à quantidade de dados que pode lidar e o número de dimensões que consegue suportar.

A tendência que existe hoje é que estas duas tecnologias convirjam para um ambiente híbrido com características relacionais e multidimensionais. Para um ambiente de data warehouse mais robusto, que contenha não só dados sumarizados mas também dados detalhados, há espaço para ambas as tecnologias. Para os dados que se encontram

no menor nível de granularidade, pode-se aplicar o modelo relacional, enquanto que para os dados sumarizados/agregados a melhor opção é o modelo dimensional. O estudo de caso abordado neste trabalho é um exemplo de como estas tecnologias podem ser aproveitadas em um mesmo ambiente.

CAPÍTULO 5

ESTUDO DE CASO
MODELO UNIVERSIDADE

Para examinarmos na prática as principais questões relativas às etapas de extração, transformação e carga dos dados durante a criação de um ambiente de data warehouse, assim como aspectos do projeto físico deste ambiente, utilizaremos o estudo de caso do ADW proposto para a Universidade Federal do Rio de Janeiro, apresentado por Vânia Soares em sua tese de mestrado - "Modelagem Incremental no Ambiente de Data Warehouse" (SOARES,1998). Em seu estudo, Vânia desenvolveu o modelo de dados deste ambiente, com base em sua proposta de modelagem incremental de dados de um ADW a partir dos dados existentes no ambiente operacional.

Neste capítulo, abordaremos as etapas de criação do ambiente, que será estruturado inicialmente sobre duas áreas de interesse, representadas por dois data marts distintos: graduação e vestibular.

O primeiro DM, que atenderá o setor de graduação, permitirá realizar:

- Avaliações sobre o desempenho dos alunos da graduação ao longo dos anos; e
- Análise das disciplinas, verificando, por exemplo, aquelas que apresentaram maiores índices de trancamento e cancelamento.

Por sua vez, o DM do vestibular possibilitará:

- Análises dos cursos mais procurados;
- Perfil dos candidatos por curso; e
- Acompanhamento das médias dos vestibulares ao longo dos anos.

É importante ressaltar que a meta deste trabalho é descrever as etapas de implementação do ambiente descrito anteriormente, com foco na extração, transformação e carga dos dados, através de uma correlação com o modelo proposto por Vânia em sua tese de mestrado.

5.1 Introdução à Modelagem Incremental

Embora não seja objetivo deste trabalho a descrição em detalhes da metodologia utilizada para a modelagem de dados do ADW, faz-se necessária uma breve introdução à mesma, pois como verificaremos a seguir, as etapas de construção do ambiente se baseiam fortemente na maneira como o modelo foi gerado.

Em sua tese "Modelagem Incremental no Ambiente de Data Warehouse", Vânia Soares propôs diretrizes para a modelagem do ADW de forma incremental, com o emprego da abordagem "bottom-up" (DM→ DW). Essas diretrizes permitem a especificação de DM e do DW, a partir do modelo corporativo, através das seguintes fases (SOARES, 1998):

FASE A: Estudo dos Modelos Existentes

Nesta fase, os projetistas definem o escopo da análise, com base no modelo corporativo ou nos DER relacionados aos sistemas existentes. As seguintes condições devem ser atendidas para iniciar esta fase:

- área de interesse definida pelo usuário final;
- levantamento dos tipos mais comuns de análises solicitadas pelo usuário final ; e
- modelo corporativo ou DER de sistemas existentes.

FASE B: Elaboração do Pré-modelo

Durante esta fase, são realizadas as análises e a integração dos DERs dos sistemas operativos que formam a área de interesse. Neste ponto, Vânia introduz o conceito de pré-modelo, caracterizado como um DER não normalizado, com as seguintes características:

- eliminação das informações operacionais, englobando não somente os atributos, mas também as entidades empregadas apenas para o processamento normal do sistema;
- desnormalização das entidades que possam ser tratadas em conjunto e daquelas que apresentam dependência de existência;
- criação de artefatos, substituindo relacionamentos dos modelos existentes, cuja informação de interesse seja extraída no momento da atualização do DW, como um "snapshot"; e

- criação de atributos derivados que possam ser considerados para o DW.

O pré-modelo surge como um resultado da integração das diversas informações existentes nos sistemas operativos, em um nível conceitual, antes do desenvolvimento do modelo dimensional (nível lógico), como acontece quando se emprega o esquema estrela.

Neste projeto, extendemos a visão do pré-modelo introduzida por Vânia; em nossa concepção, além de servir como modelo conceitual para a modelagem das visões dimensionais dos data marts, ele funciona como um modelo relacional dos dados detalhados do ADW. Desta forma, ampliamos o poder de análise oferecido aos usuários, pois estes poderão realizar consultas que não se limitam somente às visões dimensionais disponíveis.

FASE C: Elaboração do modelo dimensional

Nesta fase, os modelos dimensionais que comporão o DM são gerados a partir do "pré-modelo" definido pela fase anterior. Estes modelos serão integrados em um modelo único para o DM, e na fase seguinte, integrados ao modelo do data warehouse.

FASE D: Integração do DM ao DW

Nesta fase, o modelo do DM será integrado ao modelo do DW. O procedimento a ser executado para a integração deve ser capaz de atualizar o modelo de dados do DW com o modelo do DM, mantendo sua consistência e integridade na passagem de um estado inicial para o estado final. Isto representa o desenvolvimento incremental do DW.

5.2 Apresentação do Modelo

A seguir, apresentaremos o modelo de dados do ADW da Universidade Federal do Rio de Janeiro, elaborado por Vânia Soares em sua tese de defesa de mestrado. O modelo de dados foi elaborado de forma incremental, sendo composto pela integração dos modelos dos dois DM em questão: o DM da graduação e o DM do vestibular. Será com base neste modelo, que serão realizadas as tarefas de extração, transformação e carga dos dados para a implementação do DW Universidade.

De fato, foram gerados dois modelos para o ADW: um modelo dimensional, resultante da integração dos DM segundo a visão Kimball, e outro, relacional, elaborado segundo a visão Inmon.

Nossa proposta para a implementação do ADW é que se utilize o modelo relacional gerado para armazenar os dados detalhados do ambiente, enquanto que o modelo dimensional suportará as visões dimensionais criadas.

O modelo relacional final resultante da integração dos DM de Graduação e Vestibular ao DW está representado na figura abaixo:

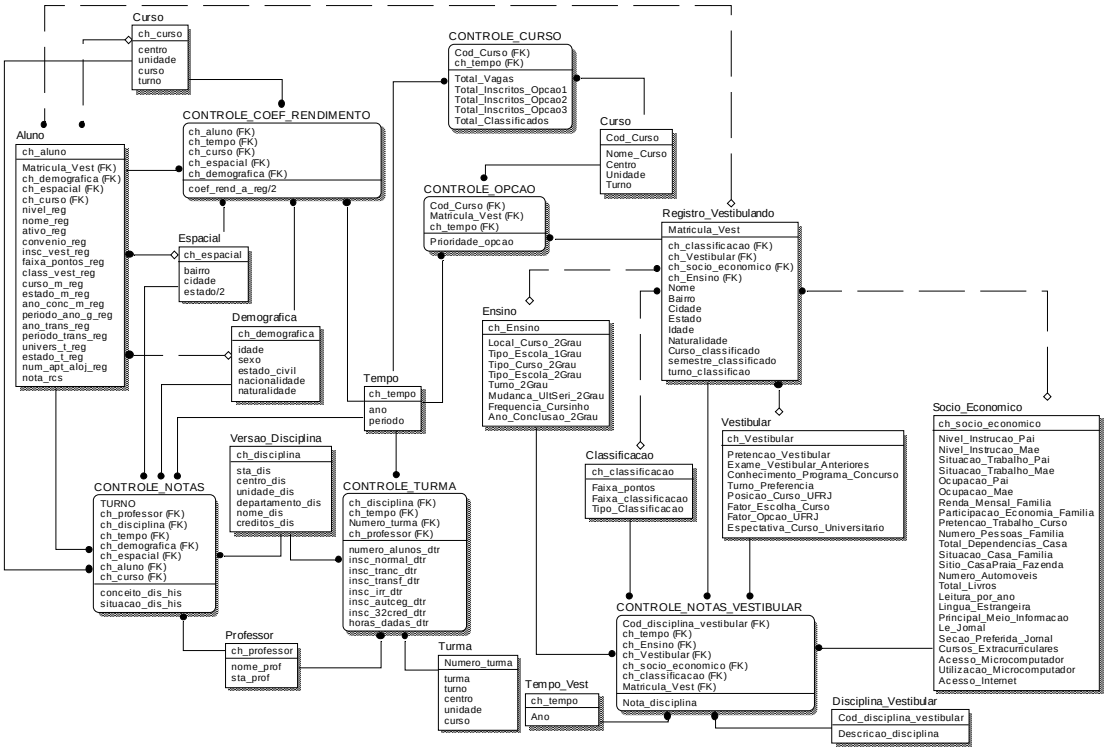

Figura 5.1- Modelo Relacional do DW Universidade

A seguir, está representado o modelo dimensional final do DW Universidade:

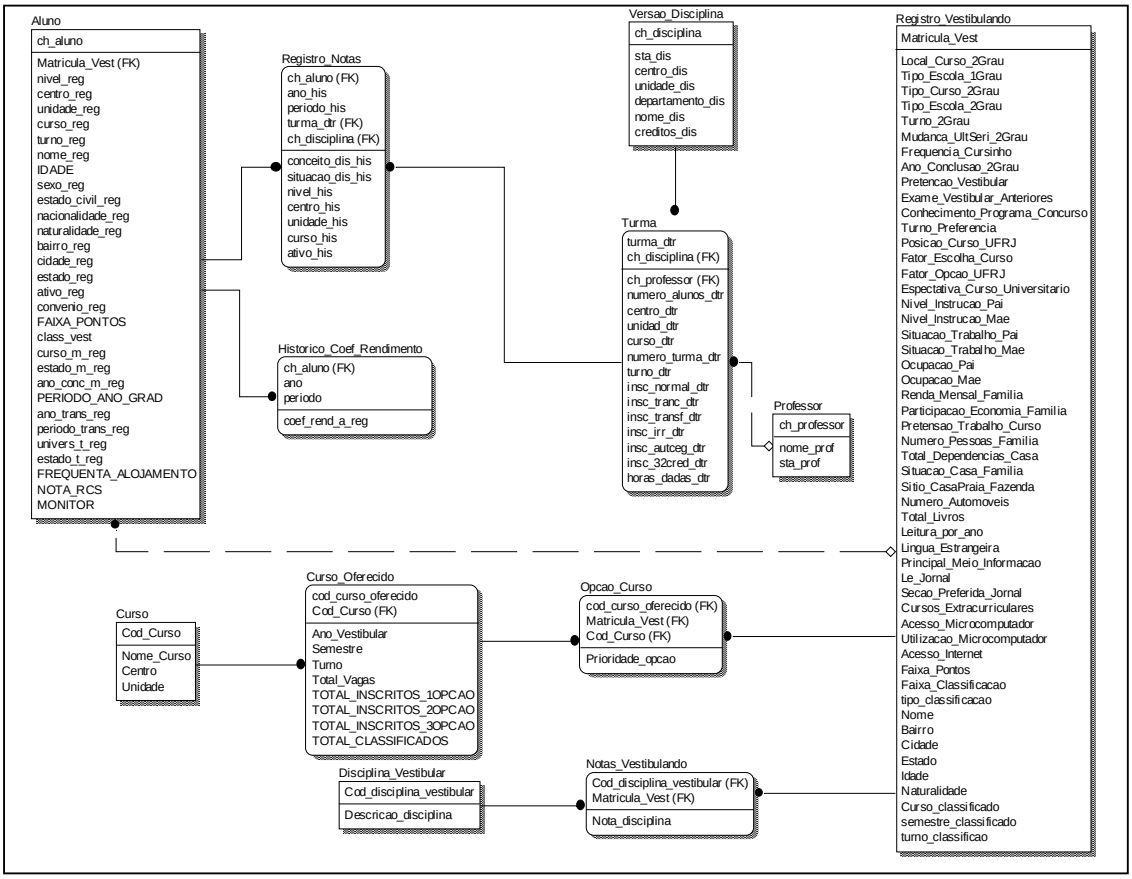

Figura 5.2 - Modelo Dimensional do DW Universidade

Em relação ao DM Graduação, os modelos acima representados contêm os seguintes fatos básicos:

- Registro de coeficiente de rendimento do aluno por ano/período;
- Registro de pontos nas disciplinas do vestibular por aluno;
- Registro de conceito e situação de aluno por disciplina em ano/período; e
- Registro de aluno inscritos, trancados, transferidos e com inscrição irregular em disciplinas por ano/período.

As seguintes visões conseguem ser estabelecidas a partir destes fatos básicos:

a) Para atender à avaliação de desempenho:

- Acompanhamento dos alunos através do coeficiente de rendimento;
- Acompanhamento com base nas notas das disciplinas; e
- Acompanhamento de cancelamento e trancamento de disciplinas.

b) Para atender à análise de disciplinas:

- Avaliação das disciplinas com maior número de trancamento/cancelamento; e
- Avaliação do desempenho das disciplinas/turmas com relação à média dos alunos.

c) Para analisar o desempenho no vestibular:

- Avaliação das notas no vestibular.

Para o DM Vestibular, identificam-se os seguintes fatos básicos:

- Registro de pontos nas disciplinas do vestibular por candidato;
- Registro de vagas fornecidas, inscritos e classificados por curso/período ao longo dos anos; e
- Registro de opções de curso por vestibulando ao longo dos anos.

Esses fatos básicos permitem estabelecer as seguintes visões:

- Acompanhamento dos vestibulandos através das médias e notas em disciplinas;
- Acompanhamento das médias por perfil de vestibulando;
- Acompanhamento das notas em disciplinas por perfil de vestibulando;
- Análise de perfil dos vestibulandos não classificados e dos classificados;
- Análise dos cursos mais procurados; e
- Avaliação das disciplinas com menores e maiores notas nos últimos anos.

5.3 Escolha do Ambiente

Como vimos no capítulo 4, é fundamental para o sucesso de implementação de um ADW que a infra-estrutura técnica seja muito bem planejada, para que não imponha limitações à evolução do data warehouse. Este planejamento deve considerar todas as etapas de desenvolvimento do ambiente, desde a seleção da abordagem de arquitetura até a disponibilização de meios de acesso aos dados para o usuário final. Como o escopo do nosso trabalho compreende apenas um subconjunto das etapas de implementação de um ADW, nossa preocupação foi a de disponibilizar um ambiente técnico que suportasse as tarefas de extração, transformação e carga dos dados nas bases de dados do ADW, desconsiderando os meios de acesso aos dados por parte dos usuários finais, além de também não se preocupar com a evolução do ADW ao longo do tempo, mesmo porque não havia subsídios suficientes para um planejamento mais refinado. Sendo assim, optamos pela simplicidade, tirando o máximo de proveito dos recursos disponíveis.

O ambiente recomendado para o desenvolvimento do ADW é a plataforma baixa, através da arquitetura cliente-servidor. O sistema gerenciador de banco de dados sugerido para armazenar os dados é o SQL Server, basicamente devido a dois fatores: disponibilidade de licença para o seu uso, e também pelo fato das informações do sistema de registro acadêmico de estudantes utilizado para obter informações para o DM Graduação, se encontrarem "espelhadas" em uma base de dados SQL Server, sob responsabilidade da área de Sistemas de Informação do NCE, tornando mais fácil a manipulação e movimentação dos dados. Os dados originais se encontram em mainframe, dificultando o acesso aos mesmos e a sua manipulação.

5.4 Processos de Extração, Transformação e Carga dos Dados

Descreveremos agora, em detalhes, cada um dos processos responsáveis pelo povoamento do ADW da Universidade Federal do Rio de Janeiro. Como veremos a seguir, estes processos possuem uma interdependência bastante alta, o que faz com que, na maioria dos casos, a fronteira entre eles não seja muito clara.

No nosso estudo de caso, todos os processos de extração, transformação e carga dos dados são representados através de scripts SQL. Em nenhum momento, empregamos uma ferramenta que nos auxiliasse na tarefa de conversão dos dados. Acreditamos que,

desta forma, obtivemos uma visão mais clara das principais dificuldades relacionadas com estes processos.

5.4.1 Extração

Em relação à extração dos dados do ambiente operacional, o primeiro passo a ser tomado é a definição das fontes de dados de onde eles serão extraídos. No nosso caso, esta identificação é direta pois existem basicamente dois sistemas legados que contemplam os dados de nosso interesse: o sistema de registro acadêmico de estudantes, sob responsabilidade do Departamento de Registro de Estudantes, de onde serão extraídos os dados para a construção do DM Graduação, e o sistema de Vestibular que, como o próprio nome deixa transparecer, contém os dados que irão compor o DM Vestibular.

Os dados de registro acadêmico estão dispostos em uma base de dados réplica SQL Server, que reflete totalmente a base de dados do sistema localizada no mainframe da instituição. A opção pelo acesso à réplica dos dados ao invés da extração direta do ambiente de grande porte se deve a três motivos: agilidade no processo, segurança e maior conhecimento da tecnologia adotada.

Como esta réplica é atualizada semestralmente, sua utilização não se restringirá à carga inicial do ambiente, podendo ser usada nas cargas subseqüentes (Isto vem ao encontro de nosso interesse, que é a realização de cargas semestrais, para a atualização de dados relativos ao período anterior).

Os dados do Vestibular também se encontram em mainframe, porém não estão armazenados em uma base de dados. Eles estão contidos, na forma de arquivos, em um sistema desenvolvido em Cobol. Estes dados só estão disponíveis uma vez por ano, após a conclusão do processo de Vestibular. Sendo assim, a extração dos dados deste sistema será realizada anualmente, no período seguinte ao término do Vestibular.

Com o objetivo de facilitar, além de padronizar, os processos de transformação dos dados, criamos uma base de dados no SQL Server para onde serão carregados os dados de Vestibular contidos nos arquivos. Desta forma, conseguimos aplicar facilmente os scripts SQLs de transformação dos dados sobre esta base.

Além das duas fontes de dados já mencionadas, utilizamos uma base de dados externa que é o cadastro de endereçamentos postais (CEPs) fornecido pelos Correios. Esta base só é necessária para se realizar o povoamento da mini-dimensão demográfica, como veremos mais adiante.

Após tomada a decisão de quais serão as fontes de dados a serem utilizadas, basta definir qual a estratégia de extração que será adotada. No nosso caso, não há dúvidas, pois só pode ser considerada a extração total, já que será dada a primeira carga de dados do ambiente. Em relação ao nosso estudo de caso, para que uma análise completa consiga ser realizada sobre o DW Universidade, optou-se por realizar a extração de dados correspondentes aos últimos cinco anos. Assim, consegue-se realizar desde o primeiro momento, o ciclo completo de acompanhamento de um conjunto de alunos, ou seja, analisar seus dados desde a realização do Vestibular até a conclusão de seus cursos.

5.4.2 Transformação

Com toda a certeza, podemos afirmar que esta etapa é a essência do data warehouse. É através dela que, de fato, os dados deixam de ser apenas dados, e passam a constituir informações relevantes para análise. É neste momento que serão implementadas todas as questões discutidas durante a fase de modelagem do data warehouse. Por esta razão, é fundamental que ela seja gerida com o máximo de qualidade, para que possa contribuir efetivamente para o sucesso da implementação do ADW.

Antes de enveredarmos pelos principais passos do processo de transformação de dados, é necessário descrever melhor o modo como a arquitetura do ambiente foi implementada. Para compor o ADW e suportar os processos ETL, foram criados dois ambientes de dados, chamados respectivamente de *staging* e produção. O primeiro representa uma área de dados intermediária, que irá conter tabelas auxiliares e dados necessários para a realização das transformações. O segundo é onde estão as tabelas que compõem o modelo do data warehouse, armazenando os dados já tratados e no qual serão realizadas todas as consultas dos usuários do ADW.

Como vimos anteriormente, nem sempre é clara a fronteira entre os processos de extração, transformação e carga dos dados em um DW. Há casos onde em um único comando SQL se consegue realizar a extração dos dados de suas fontes, efetuar as transformações necessárias, e ainda carregá-las em ambiente de produção. Isto é comum para os processos de carga das tabelas dimensionais pequenas. Porém, isto representa apenas uma pequena parte dos processos. Em geral, um processo de transformação requer diversos passos, necessitando muitas vezes de tabelas auxiliares e de consulta a dados de diferentes tabelas para o tratamento de diversos atributos. Por esta razão, optamos pela criação de uma área de dados intermediária entre as fontes de dados e o ambiente de produção, onde todos os artifícios que suportam os processos de transformação possam ser usados, em um ambiente controlado, sem que fosse necessário utilizar o próprio ambiente de produção para realizar esses tratamentos. Desta forma, a produção conterá somente as tabelas que compõem o modelo do DW, ou seja, aquelas de interesse para o negócio.

Podemos afirmar que o ambiente de *staging* é volátil em relação aos dados que contém. Todas as tabelas que possui só são necessárias no momento em que estão sendo processadas as transformações; após as mesmas, todos os dados destas tabelas são apagados, para que o ambiente esteja pronto para a próxima carga do ADW.

Os modelos de dados dos dois ambientes, assim como os scripts SQL de geração dos mesmos, se encontram descritos nos Anexos 1 e 2 respectivamente.

A seguir, serão descritos os principais passos do processo de transformação de dados necessários para a implementação do ADW Universidade. Para uma melhor compreensão, iremos descrever as tarefas separadamente para os DM Graduação e Vestibular, traçando um paralelo com a modelagem incremental proposta por Vânia (SOARES, 1998).

DM Graduação

Para a construção deste DM, iremos tratar os dados das tabelas que compõem o sistema de registro acadêmico de estudantes. Na figura abaixo, temos uma representação simplificada do DER em questão. As entidades e relacionamentos em cores são os que

fazem parte do escopo da análise. Em azul, estão as principais entidades e, em vermelho, as consideradas de interesse.

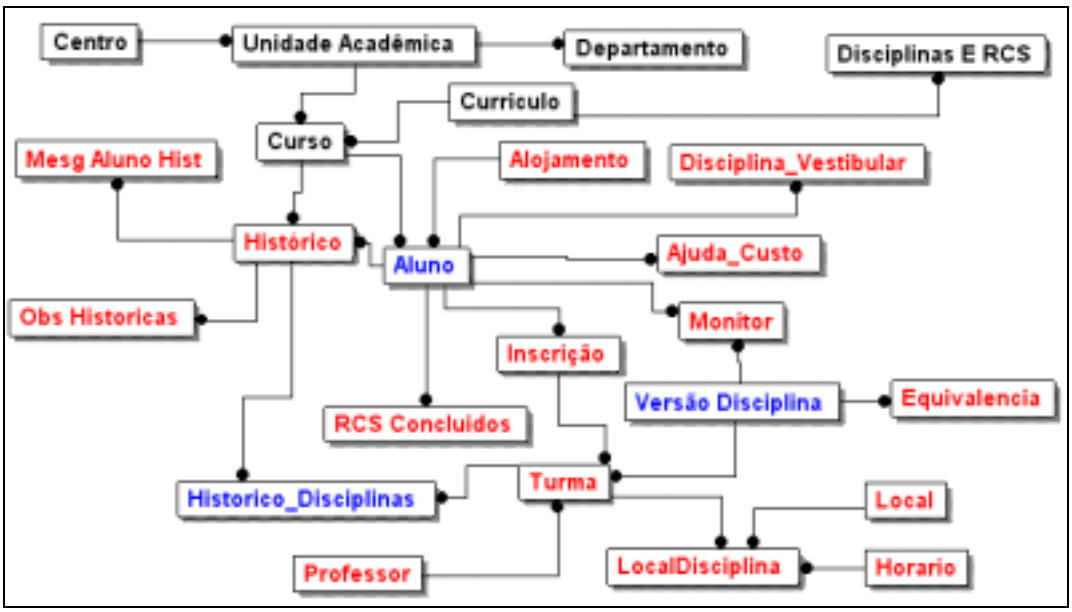

Figura 5.3 – DER do Sistema de Registro Acadêmico

O primeiro passo a ser realizado é a exclusão de todas as tabelas e atributos que são desnecessários, ou seja, não apresentam informações de interesse, de acordo com o modelo de dados do DW gerado. Esta tarefa é trivial, pois basta não realizar a extração destas informações do ambiente operacional.

Dentre as tabelas desconsideradas, podemos citar:

- *Equivalência:* que contém informações operativas de disciplinas;
- *Histórico_Obs e Histórico_Mensagem:* contém informações operativa de histórico de alunos;
- *Local_disciplina, Horário e Sala:* contém informações operativa de turma.

O passo seguinte descrito na metodologia da modelagem incremental é a desnormalização de relações. Para o DM Graduação, só identificamos um caso onde isto

acontece: a tabela *Historico* com a tabela *Historico_Disciplina*. Esta desnormalização é válida, pois é possível observar que compartilham a chave primária, apresentam os dados normalmente juntas e possuem um padrão de inserção semelhante.

Para efetuar a desnormalização, foi criada a tabela *Registro_Notas* no ambiente de *staging*, como podemos verificar no comando SQL descrito abaixo:

```
CREATE TABLE staging.registro_notas (
ch_aluno            char(10),
ano_his             char(4),
periodo_his         char(1),
turma_dtr           char(3),
ch_disciplina       char(6),
conceito_dis_his    char(3),
situacao_dis_his    char(15),
nivel_his           char(15),
centro_his          char(15),
unidade_his         char(15),
curso_his           char(15),
ativo_his           char(10)
);

INSERT INTO staging.registro_notas
SELECT
a.numero_his+'-'+a.dv_his_g2,
a.ano_his,
a.periodo_his,
b.turma_dis_his,
b.part_alfa_dis_his+'-'+b.parte_num_dis_his,
b.conceito_dis_his,
b.situacao_dis_his,
a.nivel_his,
a.centro_his,
a.unidade_his,
a.curso_his,
a.ativo_his,

FROM fonte.historico a, fonte.historico_disciplina b
WHERE       a.numero_his = b.numero_his
AND   a.dv_his = b.dv_his
AND   a.ano_his = b.ano_his;
```

A criação da tabela *Registro_Notas* no ambiente *staging* é necessária, pois serão realizadas mais transformações sobre a mesma antes da carga de suas informações no ambiente de produção. Como podemos observar acima, o comando pressupõe que uma

conexão com o ambiente operacional, onde estão situadas as fontes de dados, já tenha sido estabelecida. Como veremos no processo de carga do data warehouse, desta mesma forma deverá ser estabelecida uma conexão entre a área de *staging* e o ambiente de produção.

De acordo com o modelo de dados, foram definidas três categorias, todas elas para a tabela *Aluno*: idade, pontuação no Vestibular e período quando ocorreu a graduação. Estes atributos, representados respectivamente por *IDADE, FAIXA_PONTOS_VEST e PERIODO_ANO_GRAD,* necessitam ser criados, para conter as informações categorizadas, substituindo assim os atributos *DIA_NASC, MES_NASC, ANO_NASC, PONTOS_VEST* e *DIA_GRAD, MÊS_GRAD* e *ANO_GRAD,* presentes na tabela-fonte.

Para a definição das categorias, a tabela *Aluno* já deve ter sido criada no ambiente de *staging* contendo os novos atributos. A seguir, apresentamos as regras de criação de cada categoria.

Categoria IDADE

A idade é calculada a partir dos atributos *DIA_NASC*, *MES_NASC* e *ANO_NASC*, subtraindo da data corrente a respectiva data.

IDADE = TODAY() - TODATE (DIA_NASC+'/'+MES_NASC+'/'+ANO_NASC);

Categoria FAIXA_PONTOS_VEST

Esta categoria é definida utilizando a seguinte regra:

PONTUAÇÃO INFERIOR A 5000 : *PONTOS_VEST*< 5000;

PONTUAÇÃO ENTRE 5000 e 7000: *PONTOS_VEST* >= 5000 e < 7000;

PONTUAÇÃO ENTRE 7000 e 8000: *PONTOS_VEST* >= 7000 e < 8000; e

PONTUAÇÃO ACIMA 8000 : *PONTOS_VEST* >= 8000.

Sua implementação é realizada através do processamento de uma sequência de comandos UPDATE que irão atualizar o atributo *FAIXA_PONTOS_VEST* para cada uma das faixas. Abaixo, está representado o comando SQL executado para a faixa de pontos

entre 5000 e 7000:

```
UPDATE staging.aluno
FROM fonte.aluno
SET  faixa_pontos_vest = 'PONTUACAO ENTRE 5000 e 7000'
WHERE  fonte.aluno.numero_reg = substr(staging.aluno.ch_aluno,1,8)
AND   fonte.aluno.dv_reg     = substr(staging.aluno.ch_aluno,10,1)
AND   fonte.pontos_vest_reg >= 5000
AND   fonte.pontos_vest_reg < 7000;
```

Categoria PERIODO_ANO_GRAD

O cálculo do período e ano é realizado a partir dos atributos *MÊS_GRAD* e *ANO_GRAD*, de acordo com a regra descrita abaixo:

Se *MES_G_REG* >= 7 então
PERIODO_ANO_GRAD = 2PERIODO + ANO_GRAD
Senão PERIODO_ANO_GRAD = 1PERIODO + ANO_GRAD

O novo atributo é preenchido através da execução de dois comandos UPDATE, um para cada período. A seguir, está descrito o SQL correspondente ao 1º período:

```
UPDATE staging.aluno
FROM fonte.aluno
SET periodo_ano_grad ='1PERIODO'+CHR(fonte.aluno.ano_g_reg)
WHERE        fonte.aluno.numero_reg = substr(staging.aluno.ch_aluno,1,8)
AND   fonte.aluno.dv_reg     = substr(staging.aluno.ch_aluno,10,1)
AND   fonte.aluno.ano_g_reg < 7;
```

A etapa seguinte à definição das categorias é a criação dos artefatos, que consistem em recursos dos projetistas para transformar um relacionamento do ambiente

operacional em uma informação de interesse no ADW. Esta informação é capturada e armazenada como um atributo, com a situação em vigor no momento da extração dos dados do ambiente operacional ("snapshot").

Foi identificada a necessidade de criação de quatro artefatos para a tabela *Aluno*. Eles são os seguintes: *FREQUENTA_ALOJAMENTO, RECEBE_AJUDA, MONITOR* e *NOTA_RCS,* criados respectivamente a partir das tabelas *Alojamento, Ajuda_Custo, Monitor* e *RCS_Concluido.*

Uma vez que os atributos já tenham sido criados na tabela *Aluno* localizada na área de *staging,* seu preenchimento é extremamente simples. Para os artefatos *FREQUENTA_ALOJAMENTO, RECEBE_AJUDA* e *MONITOR* basta atualizá-los com os valores “Sim” ou “Não” de acordo com as seguintes regras:

FREQUENTA_ALOJAMENTO:
Se Aluno.NUM_ALOJ_REG > 0 **Sim**, senão **Não**

RECEBE_AJUDA:
Se existe *Ajuda_Custo* para aluno então **Sim**, senão **Não**

MONITOR:
Se existe *Monitor* para aluno então **Sim**, senão **Não**

Abaixo, temos a tradução deste tipo de regra para um comando SQL, para o caso do artefato *FREQUENTA_ALOJAMENTO*:

```
UPDATE staging.aluno
SET frequenta_alojamento = 'Sim'
WHERE ch_aluno in (
                SELECT numero_aloj+"-"+dv_aloj
                FROM fonte.alojamento );
```

Já o artefato *NOTA_RCS* será atualizado com o conteúdo armazenado no atributo

NOTA_RCS da tabela *RCS_Concluido*, como indica o comando SQL abaixo:

```
UPDATE staging.aluno
FROM fonte.aluno
SET staging.aluno.nota_rcs  = fonte.aluno.nota_rcs
WHERE        fonte.aluno.numero_rcs = substr(staging.aluno.ch_aluno,1,8)
AND   fonte.aluno.dv_rcs     = substr(staging.aluno.ch_aluno,10,1);
```

Até este ponto, as operações de transformação realizadas corresponderiam ao seguinte DER:

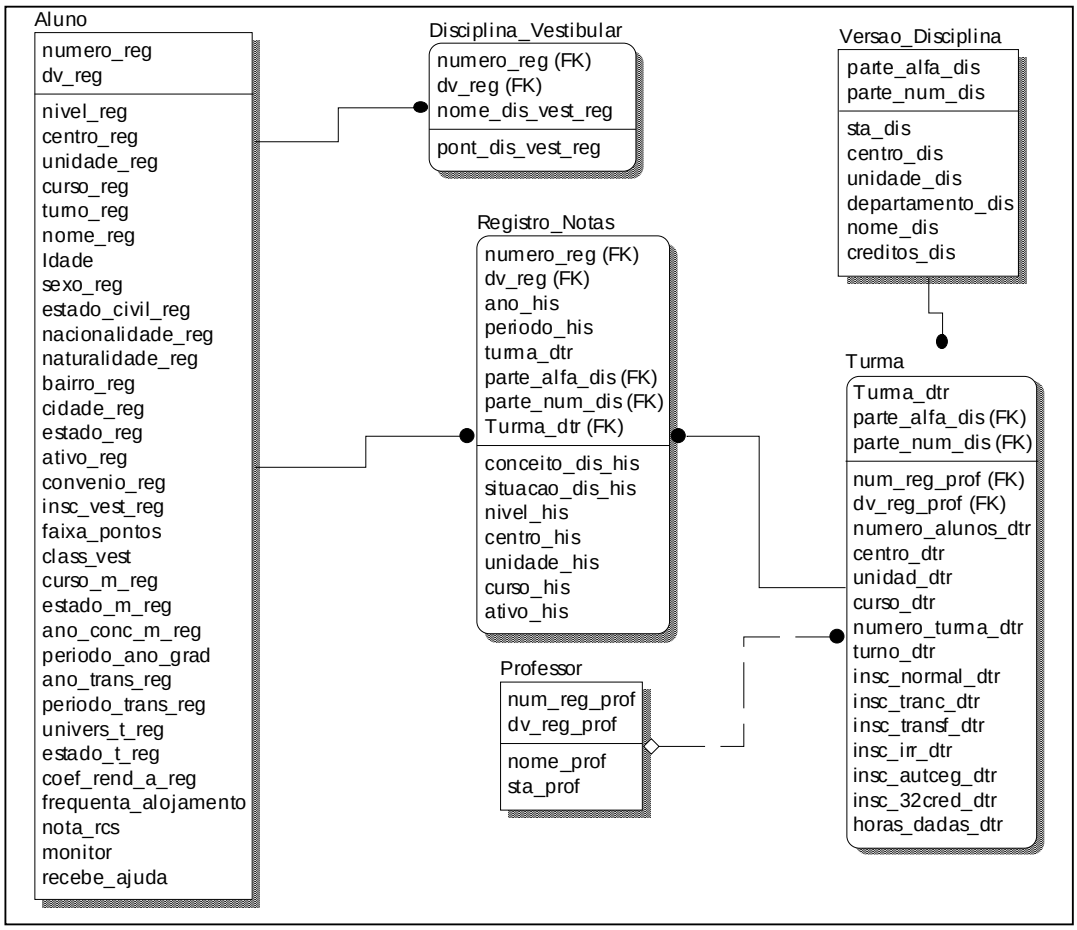

Figura 5.4 – DER resultante após a realização das primeiras transformações

Embora as chaves das tabelas acima não sejam reutilizadas no ambiente operacional, iremos criar novas chaves, mais simples, para facilitar o acesso aos dados.

Desta forma, criaremos os atributos *CH_ALUNO, CH_PROFESSOR* e *CH_DISCIPLINA*, como chaves para as entidades *Aluno, Professor* e *Versao_Disciplina*, respectivamente. As regras para a definição destas novas chaves estão descritas a seguir:

Aluno.CH_ALUNO:

Aluno.NUMERO_REG +"-"+ Aluno.DV_REG

Professor.CH_PROFESSOR:

Professor.NUM_REG_PROF +""+ Professor.DV_REG_PROF

Versao_Disciplina.CH_DISCIPLINA:

Versao_Disciplina.PARTE_ALFA_DIS + "" +
Versao_Disciplina.PARTE_NUM_DIS

Durante a análise de periodicidade dos atributos, na fase de modelagem do ADW, foi identificado que o atributo *COEF_REND_A_REG* pertencente a entidade *Aluno* requer mapeamento, visto que é periodicamente atualizado. Isto acarreta a criação da tabela *Historico_Coef_Rendimento,* que irá se relacionar com a tabela *Aluno.* Com isto, deve-se excluir o atributo referente a coeficiente de rendimento da tabela *Aluno.* Como as informações de coeficiente de rendimento variam a cada período, é necessário que a tabela *Historico_Coef_Rendimento* contenha as informações de ano e período, como podemos verificar no comando SQL de criação desta tabela:

```
CREATE TABLE staging.historico_coef_rendimento (
ch_aluno        char(10),
ano             char(4),
periodo         char(1),
coef_rend       char(3) );
```

O próximo passo a ser realizado é o estabelecimento de padrões, valores "default" e regras de conversão para substituir códigos e abreviaturas dos atributos. As tabelas *Aluno, Versao_Disciplina, Turma e Professor* necessitam de tratamento para alguns de

seus atributos. As regras para este tratamento estão relacionadas abaixo:

Atributo	**Formato**	**"default"**	**Regras para Conversão**
nivel_reg	char(15)	"Graduação"	**1,2,3 → graduação,** **4 → extensão,** **5 → aperfeiçoamento,** **6 → especialização,** **7 → mestrado,** **8 →doutorado,** **9→ pós-doutorado**
centro_reg	char(15)	Nulo	**Substituir código pela descrição**
unidade_reg	char(15)	Nulo	**Substituir código pela descrição**
curso_reg	char(15)	Nulo	**Substituir código pela descrição**
sexo_reg	char(1)	"M"	**"0" → "M"** **"1" → "F"**
nacionalidade_reg	char(15)	Nulo	**1 → brasileiro,** **2 → Naturalizado,** **3 → Estrangeiro**
naturalidade_reg	char(15)	Nulo	**Substituir código pela descrição**
ativo_reg	char(10)	"Ativo"	**"A" → ativa,** **"T" → trancada,** **"C"→ cancelada**
curso_m_reg	char(20)	Nulo	**Substituir código pela descrição**
univers_t_reg	char(20)	Nulo	**Substituir código pela descrição**
Nota_RCS	char(02)	Nulo	-----

Tabela 5.1 – Padrões, valores "default" e Regras de conversão para Aluno

Atributo	**Formato**	**"default"**	**Regras para a Conversão**
centro_dis	char(15)	Nulo	**Substituir código pela descrição**
unidade_dis	char(15)	Nulo	**Substituir código pela descrição**
curso_dis	char(15)	Nulo	**Substituir código pela descrição**

Tabela 5.2 – Padrões, valores "default" e Regras de conversão para Versao_Disciplina

Atributo	**Formato**	**"default"**	**Regras para a Conversão**
centro_dtr	char(15)	Nulo	**Substituir código pela descrição**
unidade_dtr	char(15)	Nulo	**Substituir código pela descrição**
curso_dtr	char(15)	Nulo	**Substituir código pela descrição**

Tabela 5.3 – Padrões, valores "default" e Regras de conversão para Turma

Entidade *Professor*			
Atributo	**Formato**	**"default"**	**Regras para a Conversão**
sta_prof	char(15)	"Ativa"	**"A" → ativo,** **"P" → aposentado,**

			"L" → licenciado, **"F" → falecido**

Tabela 5.4 – Padrões, valores "default" e Regras de conversão para Professor

Por constituir uma lista extensa, a relação dos *scripts* SQL de tratamentos destes atributos está descrita no Anexo 2.

A figura 5.5 apresenta o resultado das transformações realizadas até este instante. Na verdade, ela representa o pré-modelo descrito por Vânia Soares em sua proposta de metodologia da modelagem incremental. Para o nosso estudo de caso, ele constitui o modelo relacional que contém os dados detalhados do ambiente.

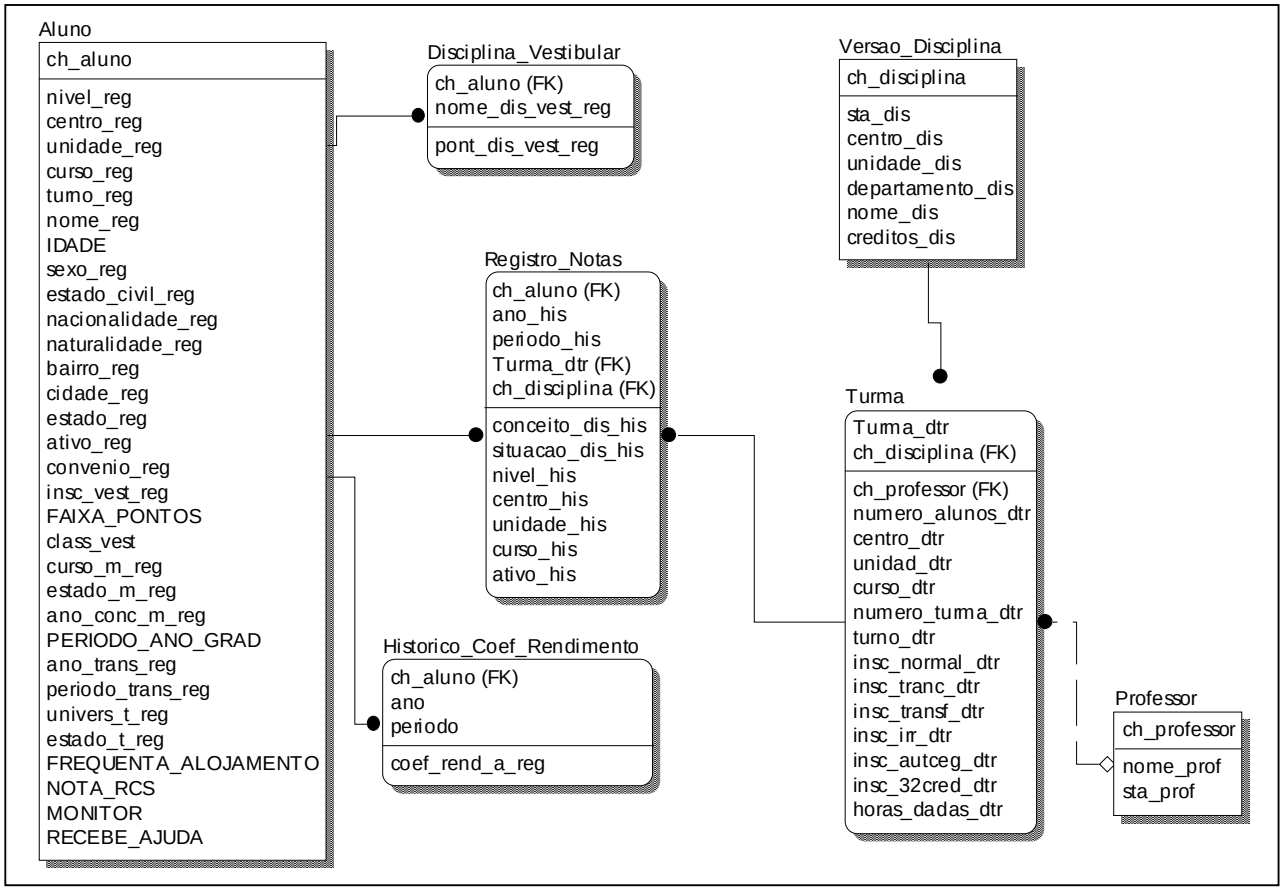

Figura 5.5 – DER do ambiente relacional correspondente ao DM Graduação

A partir de agora, todas as transformações visam à implementação do modelo de dados dimensional do ADW. Sendo assim, o foco será dado na geração dos modelos

dimensionais de cada um dos fatos básicos estabelecidos para este DM, descritos a seguir:

- Registro de coeficiente de rendimento do aluno por ano/período;
- Registro de pontos nas disciplinas do vestibular por aluno;
- Registro de conceito e situação de aluno por disciplina em ano/período; e
- Registro de aluno inscritos, trancados, transferidos e com inscrição irregular em disciplinas por ano/período.

Abaixo, temos o modelo dimensional correspondente ao primeiro fato a ser gerado, que é o registro de coeficiente de rendimento do aluno por ano/período:

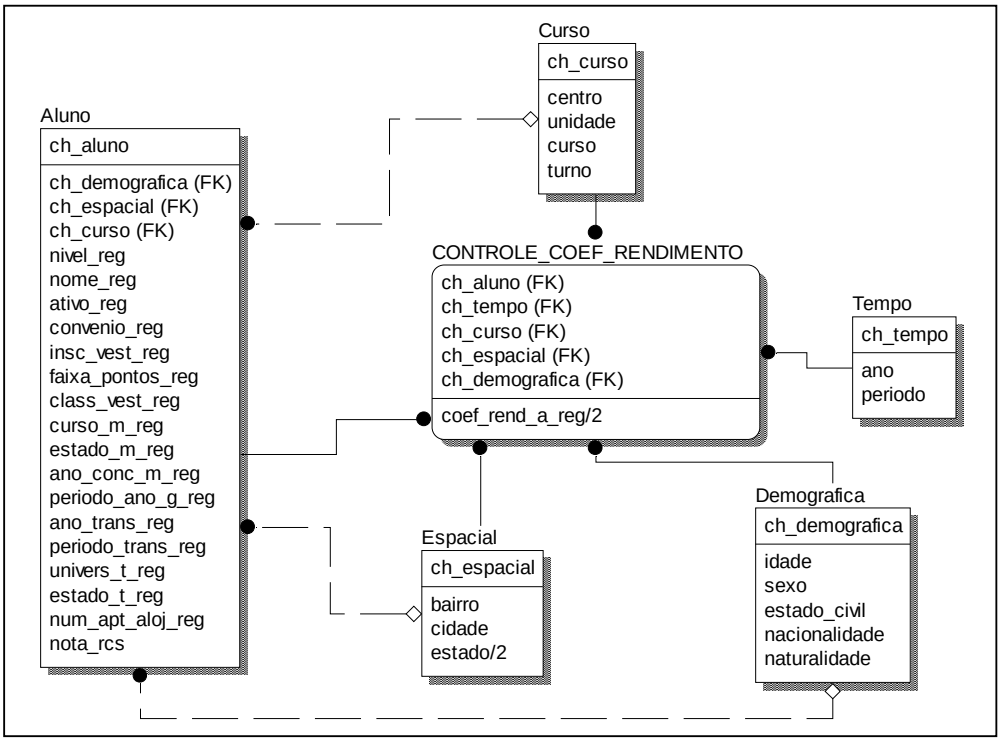

Figura 5.6 – Modelo Dimensional para o fato CONTROLE_COEF_RENDIMENTO

Analisando o modelo acima, verificamos que foram criadas três minidimensões, a partir dos atributos pertencentes à dimensão **ALUNO: CURSO, DEMOGRÁFICA E ESPACIAL**. A criação destas minidimensões garantirá um melhor desempenho nas consultas, além de uma melhor visualização dos dados contidos na dimensão. A primeira

contém atributos referentes à identificação do curso selecionado pelo aluno, a segunda apresenta informações demográficas (como idade, sexo e estado civil) e a terceira contém informações de localidade.

O preenchimento da minidimensão **CURSO** é direto, pois basta selecionar os atributos necessários que estão contidos na tabela *Curso* presente no ambiente operacional. Já o povoamento da minidimensão **DEMOGRÁFICA** requer uma outra abordagem. Todos os registros que irão povoá-la serão selecionados da tabela *Aluno*, através de um comando SELECT DISTINCT que trará os valores dos atributos que compõem esta minidimensão.

Desta forma, garante-se que quaisquer que sejam as informações cadastradas para um aluno específico, existirá uma linha na minidimensão que corresponderá a esses valores. Uma outra forma de efetuar este povoamento seria através da realização de um produto cartesiano com os valores dos atributos da dimensão. Porém, para este caso, seria gerado um número muito alto de registros, o que fez com que não optássemos por esta alternativa.

A terceira e última minidimensão a ser criada para este DM é a **ESPACIAL**. Seu preenchimento demandou uma abordagem diferenciada. Uma das opções para povoá-la seria utilizar os próprios valores contidos na tabela *Aluno*. No entanto, analisando seus dados, verificamos que eles possuíam baixíssima qualidade. Foi identificado que inúmeros registros continham os mesmos valores, porém representados de forma diferentes, devido a abreviações e erros de digitação. Por esta razão, para garantir a qualidade e a consistência dos dados, optamos por utilizar as informações contidas na base de dados de endereçamento postal fornecida pelos Correios. Desta maneira, através do atributo cep_reg da tabela *Aluno*, conseguimos carregar, com a máxima qualidade possível, as informações de bairro, cidade e estado. Nos casos em que o valor do CEP presente na tabela *Aluno* era inconsistente ou nulo, utilizamos os atributos relativos ao bairro, cidade e estado nela contidos. Com esta abordagem, diminuímos bastante o número de registros que necessitaram passar por uma análise manual. Todos os registros da dimensão **ALUNO** que no ambiente operacional não possuíam estas informações cadastradas foram relacionados com um registro da minidimensão que contém o valor “Não Informado” para seus três atributos.

Uma vez extraídas as informações da dimensão **ALUNO** para compor as três minidimensões necessárias, basta agora realizar o preenchimento das informações da dimensão **TEMPO,** que estabelece a granularidade ano/período como sendo a adotada pelo ADW. Para preenchê-la, basta inserir diretamente os registros, com dados a partir do primeiro ano a ser analisado.

A última ação necessária para completar a implementação do fato básico de registro de coeficiente de rendimento do aluno por ano/período é a criação da tabela de fatos **CONTROLE_COEF_RENDIMENTO**. Esta tabela será preenchida conforme o comando SQL descrito abaixo:

```
INSERT INTO staging.dim_controle_coef_rendimento
SELECT
        a.ch_curso,
        a.ch_aluno,
        a.ch_espacial,
        a.ch_demografica,
        b.ch_tempo
        c.coef_rend_a_reg
FROM
        staging.dim_aluno a,
        staging.dim_tempo b,
        staging.historico_coef_rendimento c
WHERE
        c.ano = b.ano
AND   c.periodo = b.periodo
AND   a.ch_aluno = c.ch_aluno;
```

O segundo fato a ser tratado é o de registro de pontos nas disciplinas do vestibular por aluno. Seu modelo dimensional pode ser observado na figura abaixo:

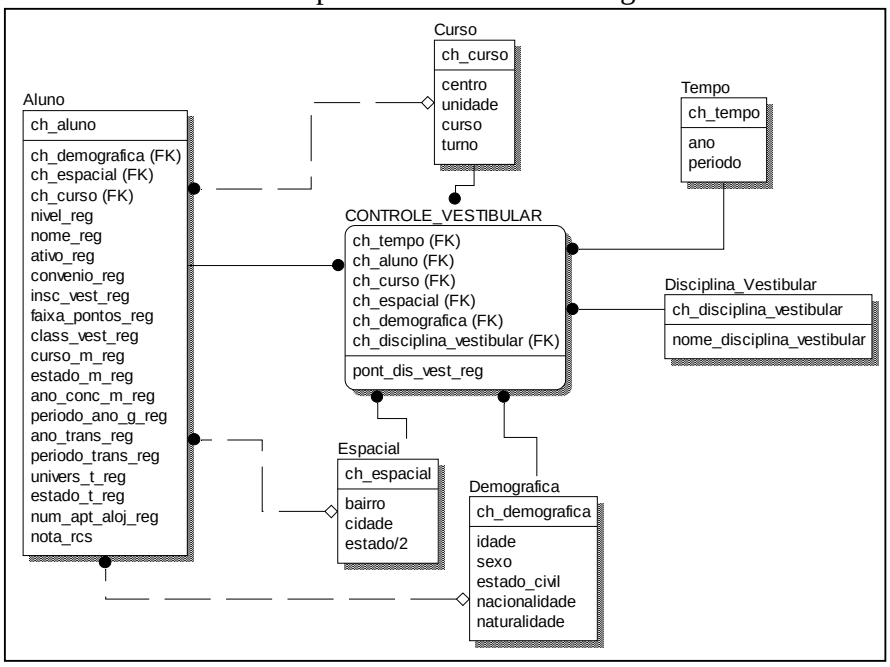

Figura 5.7 – Modelo Dimensional para o fato CONTROLE_VESTIBULAR

Como podemos observar acima, com exceção da dimensão **DISCIPLINA_VESTIBULAR**, todas as outras dimensões já foram criadas para a visão dimensional tratada anteriormente. Para preencher esta nova dimensão, basta selecionar os valores contidos no atributo *NOME_DIST_VEST_REG* da tabela *Disciplina_Vestibular* presente na base de dados fonte. Essa nova dimensão apenas será atualizada quando uma nova disciplina for criada.

O povoamento da tabela de fatos **CONTROLE_VESTIBULAR** está descrito no comando SQL abaixo:

```
INSERT INTO staging.dim_controle_vestibular
SELECT
        a.ch_aluno,
        a.ch_curso,
        a.ch_espacial,
        a.ch_demografica,
        b.ch_disciplina_vestibular,
```

```
        c.ch_tempo,
        b.pont_dis_vest_reg
FROM
        staging.dim_aluno a,   staging.disciplina_vestibular b,
        staging.dim_tempo c,  fonte.aluno d
WHERE
        a.ch_aluno = d.numero_reg+'-'+d.dv_reg
AND   c.periodo = DECODE(d.mes_u_reg,'01','1','02','1','03','1',
                '04','1','05','1','06','1','07','1','2')
AND   c.ano = d.ano_u_reg
AND   a.ch_aluno = b.ch_aluno;
```

A seguir, está representado o modelo dimensional correspondente ao terceiro fato básico deste DM - registro de conceito e situação de aluno por disciplina em ano/período:

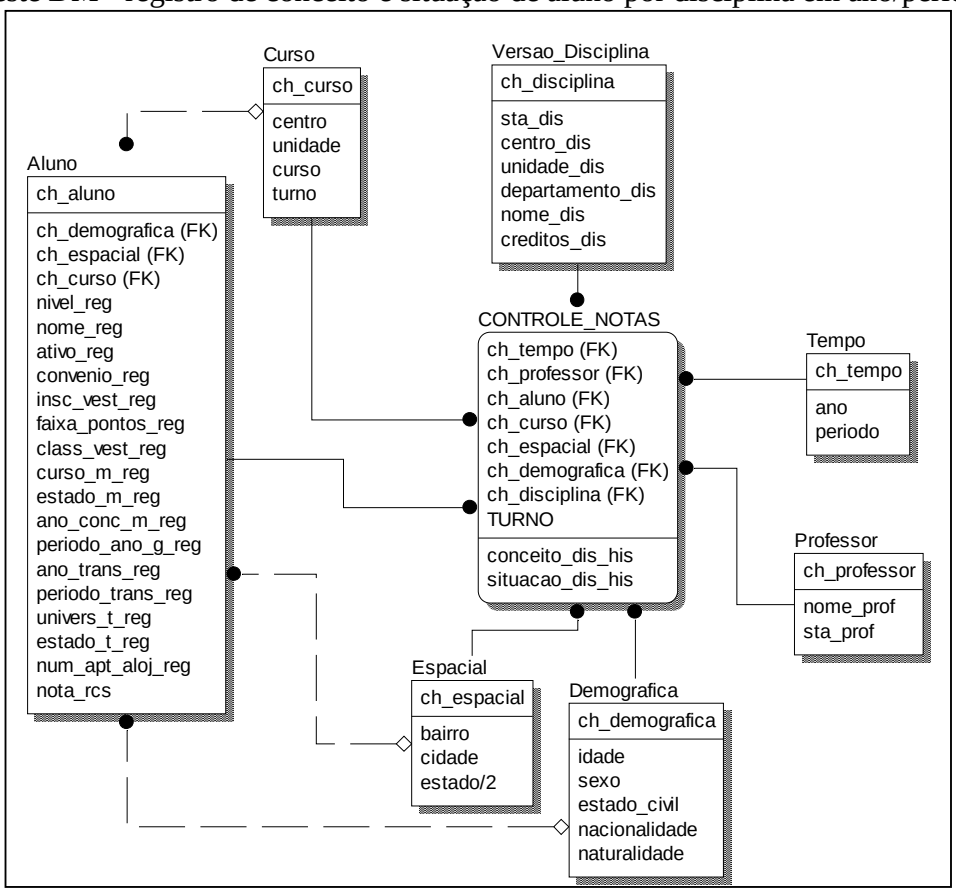

Figura 5.8 – Modelo Dimensional para o fato CONTROLE_NOTAS

Para a implementação desta visão dimensional, será criada a tabela de fatos CONTROLE_NOTAS, gerada a partir da tabela *Registro_Notas* presente no ambiente relacional do ADW.

É interessante notar a inclusão do atributo *TURNO* nesta tabela. Ele representa a dimensão descaracterizada correspondente à tabela *Turma* do ambiente relacional. Esta tabela possui um conjunto de informações que não dizem respeito a análise, e portanto não necessitam estar contidas na dimensão. São elas: *NUMERO_ALUNOS, INSC_NORMAL_DTR, INSC_TRANC_DTR, INSC_TRANSF_DTR, INSC_IRR_DTR, INSC_AUTCEG_DTR, INS_32CRED_DTR, HORAS_DADAS.* Como o único atributo da tabela *Turma* que interessa para a visão em questão é o turno, optou-se por criá-lo na tabela de fatos.

Abaixo, podemos observar o SQL de inserção de dados na tabela de fatos **CONTROLE_NOTAS** :

```
INSERT INTO staging.dim_controle_notas
SELECT
        c.ch_professor,
        a.ch_disciplina,
        d.ch_tempo,
        b.ch_demografica,
        b.ch_espacial,
        b.ch_aluno,
        c.ch_curso,
        c.turno,
        a.conceito_dis_his,
        a.situacao_dis_his
FROM
        staging.registro_notas a,
        staging.dim_aluno b,
        staging.turma c,
```

```
        staging.dim_tempo d
WHERE
        a.ch_aluno = b.ch_aluno
AND     a.ano_his = d.ano
AND     a.periodo_his = d.periodo
AND     a.turma_dtr = c.turma_dtr;
```

Para finalizar a construção do DM Graduação, resta implementar o modelo dimensional relativo ao último fato básico que o compõe: registro de alunos inscritos, trancados, transferidos e com inscrição irregular em disciplinas por ano/período.

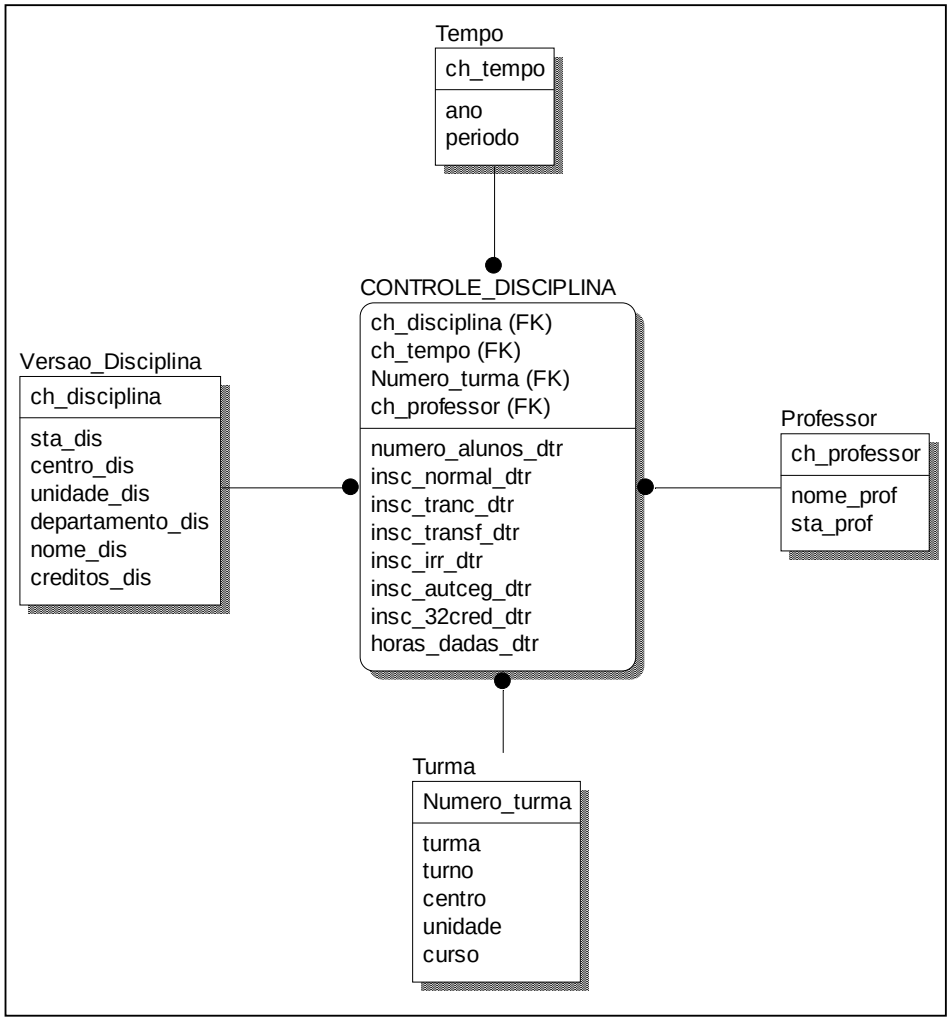

Figura 5.9 – Modelo Dimensional para o fato CONTROLE_DISCIPLINA

A tabela que servirá como base para a implementação deste fato é a tabela *Turma*. Observando o modelo acima, verificamos a criação da dimensão **TURMA** a partir dos atributos *NUMERO_TURMA_DTR, TURNO_DTR, CENTRO_DTR, UNIDADE_DTR, CURSO_DTR* presentes na tabela original. O comando SQL de preenchimento da tabela de fatos **CONTROLE_DISCIPLINA** é o seguinte:

```
INSERT INTO staging.controle_disciplina
SELECT
        a.ch_disciplina,
        b.ch_tempo,
        a.numero_turma,
        a.ch_professor,
        a.numero_alunos_dtr,
        a.insc_normal_dtr,
        a.insc_tranc_dtr,
        a.insc_transf_dtr,
        a.insc_irr_dtr,
        a.insc_autceg_dtr,
        a.insc_32cred_dtr,
        a.horas_dadas_dtr
FROM
        staging.turma a,
        staging.dim_tempo b,
WHERE
        b.ano = a.ano
AND   b.periodo = a.periodo;
```

Com isto, terminamos a implementação do DM Graduação, composto por dois ambientes: o relacional, baseado no pré-modelo, e o dimensional, através da construção das visões dimensionais relativas aos fatos básicos identificados.

DM Vestibular

Na etapa de implementação deste DM, são tratadas as informações contidas na base de dados criada a partir dos arquivos gerados pelo sistema de vestibular da Universidade. A figura abaixo representa o DER simplificado desta base:

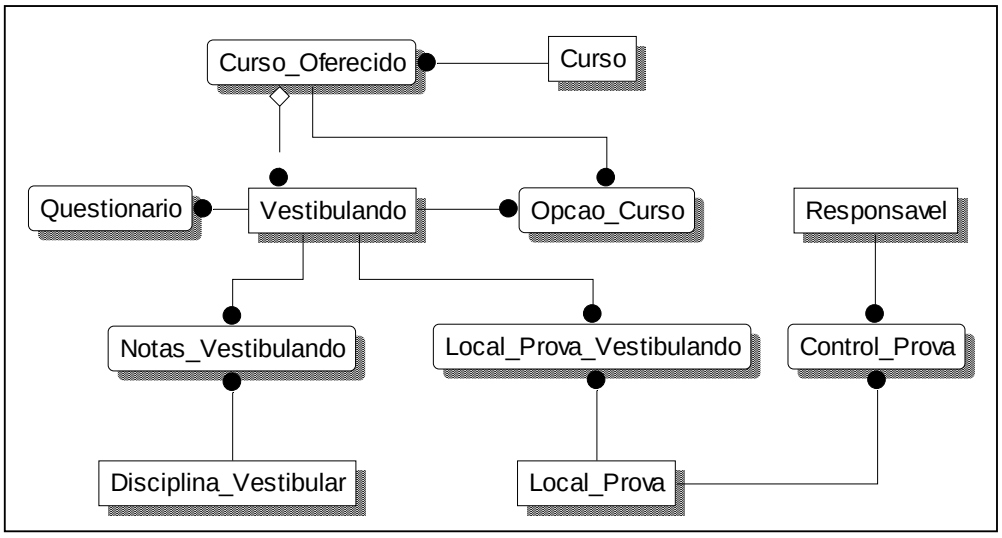

Figura 5.10 – DER simplificado da base de dados que contém os dados do Vestibular

Primeiramente, devem ser desconsideradas todas as tabelas e atributos que não são de interesse para o DW. Estas informações não deverão sequer ser extraídas do ambiente operacional. As tabelas que serão ignoradas são as seguintes:

- *Responsavel*: Informações operacionais de controle dos responsáveis em aplicar as provas;
- *Controle_Prova*, *Local_Prova* e *Local_Prova_Vestibulando*: Informações operacionais de controle do local das provas.

A primeira transformação a ser realmente realizada é a criação da tabela *Registro_Vestibulando* no ambiente de *staging* , que representa a desnormalização das tabelas *Vestibulando* e *Questionario*. Pelo fato destas tabelas compartilharem a chave primária, apresentarem os dados normalmente juntos e possuírem um padrão de inserção semelhante, esta desnormalização foi considerada válida. O comando SQL de criação desta tabela, por ser bastante extenso, se encontra descrito no Anexo 2.

A tabela *Registro_Vestibulando* apresenta as seguintes informações a serem categorizadas: data de nascimento, pontuação vestibular e classificação no vestibular. Os atributos *IDADE, FAIXA_PONTOS_VEST e FAIXA_CLASSIFICACAO* serão criados contendo as informações categorizadas e substituindo os atributos *DATA_NASCIMENTO, PONTOS_VESTIBULAR* e *CLASSIFICACAO_VESTIBULAR*.

A definição das categorias *IDADE* e *FAIXA_PONTOS_VEST* seguiu as mesmas regras utilizadas para implementar as categorias homônimas contidas no DM Graduação. Já para a definição da categoria *CLASSIFICACAO_VESTIBULAR* foi adotada a seguinte regra:

DEZ PRIMEIROS: CLASSIFICACAO_VESTIBULAR <= 10;
ENTRE DECIMO E VIGESIMO: *CLASSIFICACAO_VESTIBULAR* > 10 e <= 20; e
ACIMA DO VIGESIMO: CLASSIFICACAO_VESTIBULAR > 20

Para a implementação da regra acima será necessária uma sequência de três comandos UPDATE, um para cada faixa definida, como podemo verificar abaixo:

```
UPDATE staging.registro_vestibulando
SET     faixa_classificacao = 'DEZ PRIMEIROS'
WHERE         faixa_classificacao <= 10;

UPDATE staging.registro_vestibulando
SET     faixa_classificacao = 'ENTRE DECIMO E VIGESIMO'
WHERE         faixa_classificacao > 10
AND    faixa_classificacao <= 20;

UPDATE staging.registro_vestibulando
SET     faixa_classificacao = 'ACIMA DO VIGESIMO'
WHERE         faixa_classificacao > 20;
```

Até o momento, as transformações realizadas corresponderiam ao modelo de DER representado abaixo:

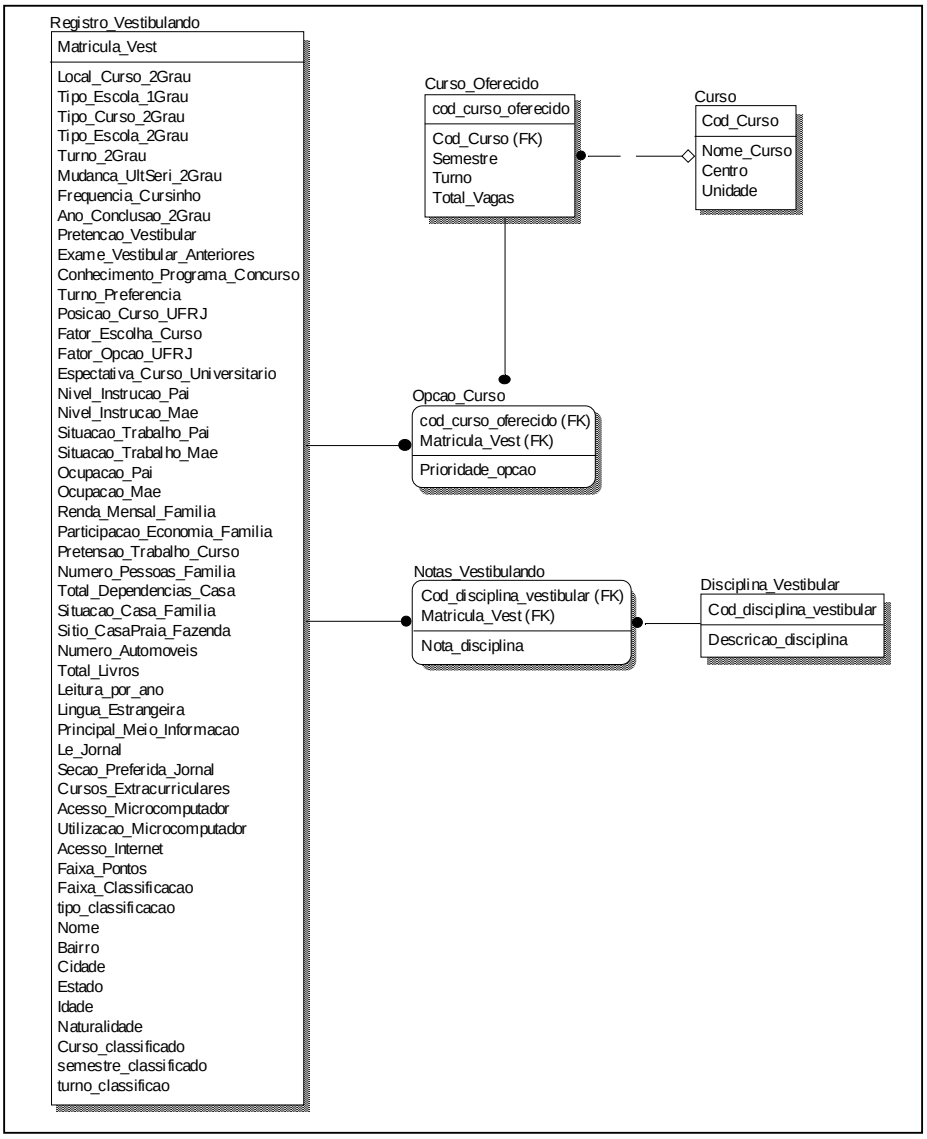

Figura 5.11 – DER resultante após a realização das primeiras transformações

Segundo o levantamento realizado durante a fase de modelagem, a matrícula do vestibulando é gerada compondo o ano do vestibular a um número seqüencial. Por esta razão, a criação de uma nova chave para esta tabela não é necessária pois não existe a possibilidade de reutilização de matrícula. Por outro lado, o atributo *COD_CURSO_OFERECIDO* da tabela *Curso_Oferecido* é formado por código do curso

e semestre em que o curso é fornecido. Neste caso, será necessária a inserção da chave tempo nesta tabela, representada pelo atributo *ANO_VESTIBULAR*.

Analisando as tabelas *Registro_Vestibulando* e *Curso*, verificamos que elas necessitam de tratamento para alguns de seus atributos. A relação destes atributos com seus respectivos tratamentos se encontra no Anexo 3.

Para atender às consultas do usuário final serão criados na tabela *Curso_Oferecido* os seguintes atributos: TOTAL_INSCRITOS_1OPCAO, TOTAL_INSCRITOS_2OPCAO, TOTAL_INSCRITOS_3OPCAO e TOTAL_CLASSIFICADOS. O cálculo destes atributos obedecem às regras abaixo:

TOTAL_INSCRITOS_1OPCAO =
Sum (Opcao_Curso onde
Opcao_Curso.COD_OPCAO_CURSO= Curso_Oferecido.COD_CURSO
e Opcao_Curso.PRIORIDADE_OPCAO = 1);

TOTAL_INSCRITOS_2OPCAO =
Sum (Opcao_Curso onde
Opcao_Curso.COD_OPCAO_CURSO= Curso_Oferecido.COD_CURSO
e Opcao_Curso.PRIORIDADE_OPCAO = 2);

TOTAL_INSCRITOS_3OPCAO =
Sum (Opcao_Curso onde
Opcao_Curso.COD_OPCAO_CURSO= Curso_Oferecido.COD_CURSO
e Opcao_Curso.PRIORIDADE_OPCAO = 3);

TOTAL_CLASSIFICADOS =
Sum (Registro_Vestibulando onde
Registro_Vestibulando.CURSO_CLASSIFICADO =
Curso_Oferecido.COD_CURSO e
Registro_Vestibulando.SEMESTRE_CLASSIFICADO =
Curso_Oferecido.SEMESTRE);

Os comandos SQL de criação dos mesmos se encontram no Anexo 2.

Com as transformações realizadas até este momento, findamos a implementação do ambiente relacional do DM Vestibular. Mais adiante veremos qual a melhor forma de

integrá-lo com o DM Graduação, e assim, constituir o ambiente relacional do ADW Universidade como um todo.

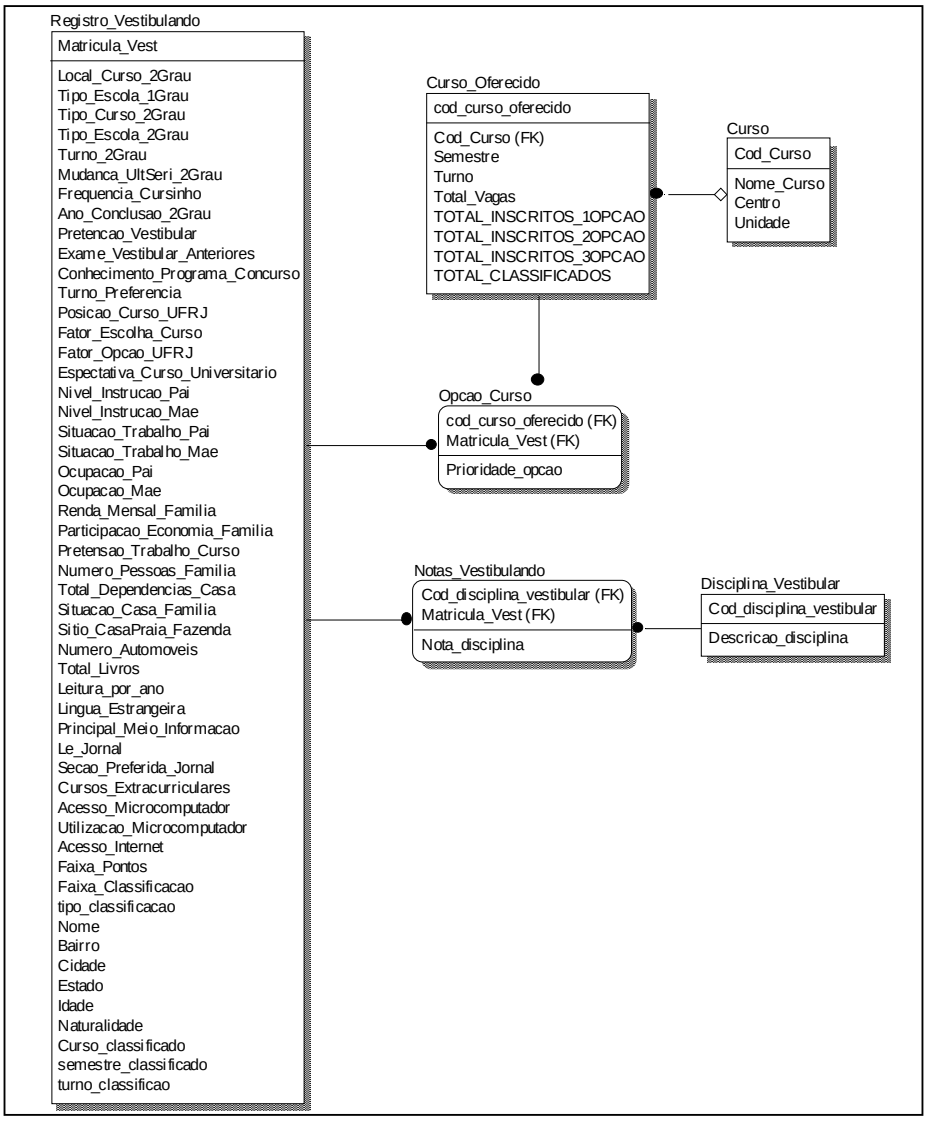

Figura 5.12 – DER do ambiente relacional correspondente ao DM Vestibular

Uma vez concluída a construção do ambiente relacional do DM, devemos partir para a realização das transformações que originarão o modelo dimensional do mesmo. Desta forma, as ações serão direcionadas para a geração dos modelos dimensionais de cada um dos fatos básicos estabelecidos para este DM, descritos a seguir:

- Registro de pontos nas disciplinas do vestibular por candidato;
- Registro de vagas fornecidas, inscritos e classificados por curso/período ao longo dos anos; e
- Registro de opções de curso por vestibulando ao longo dos anos.

Para o primeiro fato básico, registro de pontos nas disciplinas do vestibular por candidato, devemos implementar o seguinte modelo:

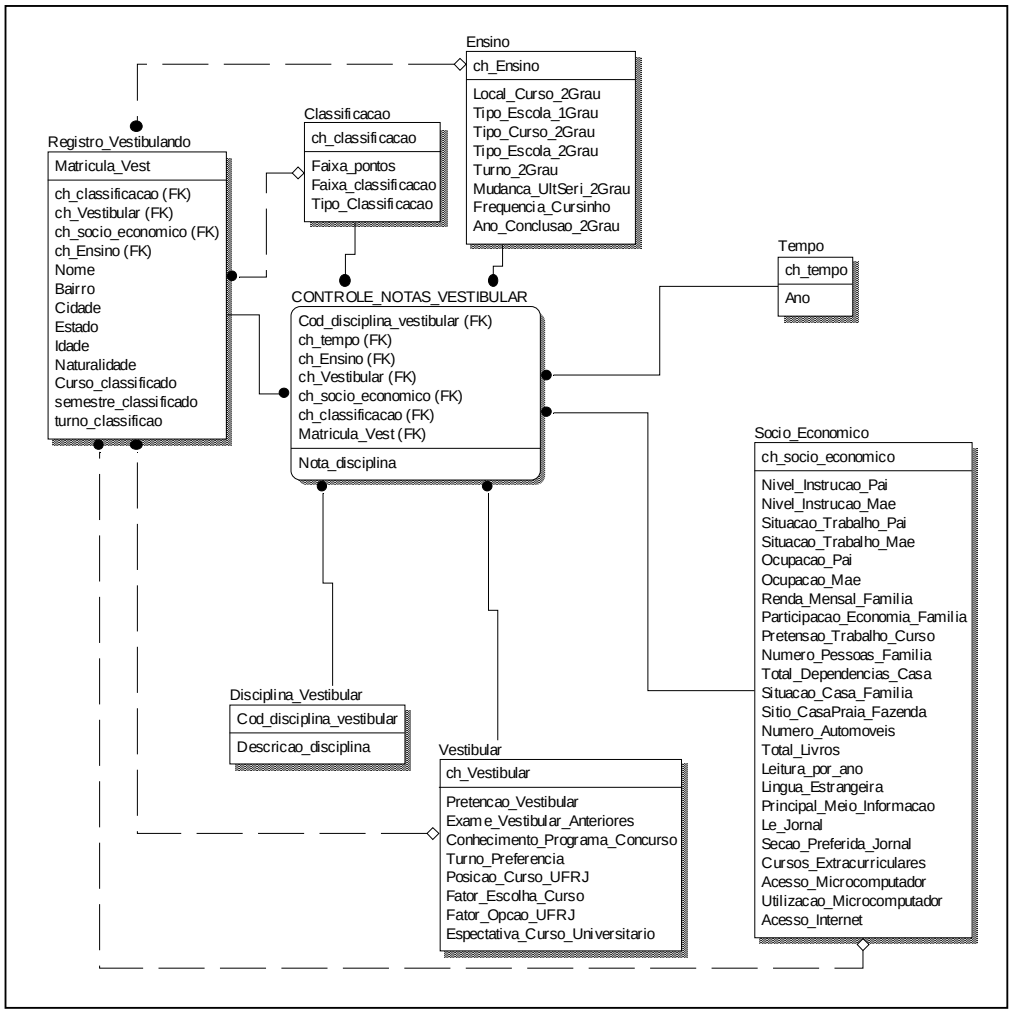

Figura 5.13 – Modelo Dimensional para o fato CONTROLE_NOTAS_VESTIBULAR

Observando o modelo acima, verificamos que a tabela *Notas_Vestibulando* registra as notas dos vestibulandos nas disciplinas para um vestibular. Portanto, as

informações estão registradas por ano. Desta forma, a dimensão **TEMPO** deve ser criada com o atributo *ANO*.

Notamos também que a tabela *Registro_Vestibulando* deu origem a quatro minidimensões compostas por grupos de informações afins : a primeira com atributos referentes a ensino, a segunda com as informações relacionadas ao vestibular na UFRJ, a terceira com informações sócio-econômicas e a quarta com informações referentes a classificação no vestibular. Estas minidimensões foram denominadas respectivamente de **ENSINO, VESTIBULAR, SOCIO_ECONOMICO** e **CLASSIFICACAO**.

A implementação destas minidimensões é direta, a partir do desmembramento dos atributos que compõem a tabela *Registro_Vestibulando.* Além disso, o modo de carregá-las é similar ao da minidimensão **DEMOGRAFICA** do DM Graduação, ou seja, deve ser realizado o produto cartesiano para cada um dos valores pertencentes aos domínios dos atributos que as compõe.

A outra dimensão que compõe esta visão é a **DISCIPLINA_VESTIBULAR** gerada a partir da tabela de mesmo nome do ambiente relacional.

Complementando a implementação deste fato, temos a criação da tabela de fatos **CONTROLE_NOTAS_VESTIBULAR** cujo preenchimento está representado abaixo através do seguinte comando SQL:

```
INSERT INTO staging.dim_controle_notas_vestibular
SELECT
        a.cod_disciplina_vestibular,
        c.ch_tempo,
        b.ch_ensino,
        b.ch_vestibular,
        b.ch_socio_economico,
        b.ch_classificacao,
        a.matricula_vest,
        a.nota_disciplina
FROM
        staging.notas_vestibulando a, staging.dim_registro_vestibulando b,
```

```
        staging.dim_tempo c,  fonte.opcao_curso d
WHERE       a.matricula_vest = b.matricula_vest
AND   a.matricula_vest = d.matricula_vest
AND   d.prioridade_opcao = '1'
AND   d.ano_vestibular = c.ano;
```

O fato seguinte a ser tratado é o que se refere ao registro de vagas fornecidas por curso/período ao longo dos anos, e que pode ser observado na figura abaixo:

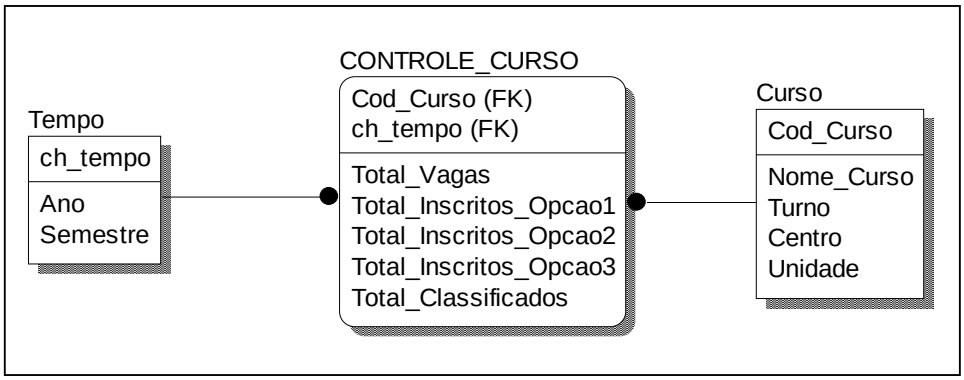

Figura 5.14 – Modelo Dimensional para o fato CONTROLE_CURSO

A tabela *Curso_Oferecido* armazena as informações por ano. Portanto, a dimensão **TEMPO** empregada, apresenta o atributo *ANO.*

Analisando a tabela *Controle_Curso,* verificamos que ele contém a chave por ano e período. Por esta razão, o atributo *COD_CURSO_OFERECIDO* será removido. Pelas características, é possível observar que o atributo *SEMESTRE* é uma informação da dimensão **TEMPO** e o atributo *TURNO* da dimensão **CURSO.** Portanto, os atributos em questão passarão a compor as respectivas dimensões.

O povoamento da tabela de fatos CONTROLE_CURSO está descrito abaixo:

```
INSERT INTO staging.dim_controle_curso
SELECT
        a.cod_curso,    b.ch_tempo,
        a.total_vagas,  a.TOTAL_INSCRITOS_1OPCAO,
        a.TOTAL_INSCRITOS_2OPCAO,   a.TOTAL_INSCRITOS_3OPCAO,
```

```
        a.TOTAL_CLASSIFICADOS
FROM    staging.curso_oferecido a,    staging.dim_tempo b
WHERE   a.ano_vestibular = b.ano;
```

Para finalizar a implementação do ambiente dimensional do DM Vestibular, resta construir a visão dimensional relativa ao fato básico de registro de opções de curso por vestibulando ao longo dos anos, representado pelo seguinte modelo:

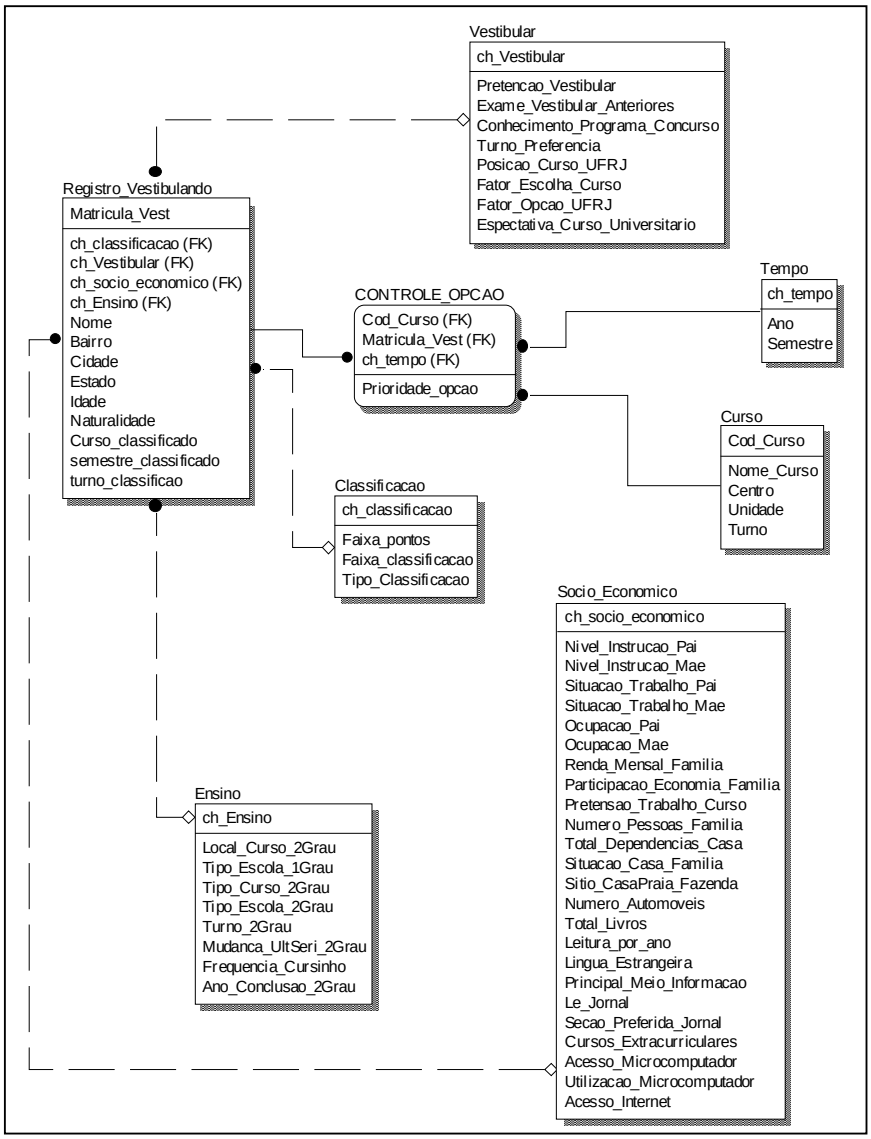

Figura 5.15 – Modelo Dimensional para o fato CONTROLE_OPCAO

A tabela *Opcao_Curso* registra as opções dos vestibulandos para cada ano de

vestibular. Portanto, a dimensão **TEMPO** deve conter o atributo *Ano.*

Abaixo, está descrito o comando SQL que preenche a tabela de fatos **CONTROLE_OPCAO**:

```
INSERT INTO staging.dim_controle_opcao
SELECT a.cod_curso_oferecido,  a.matricula_vest,
        b.ch_tempo, a.prioridade_opcao
FROM   staging.opcao_curso a, staging.dim_tempo b, staging.curso_oferecido c
WHERE  a.cod_curso_oferecido = c.cod_curso_oferecido
AND    a.cod_curso = c.cod_curso AND      c.ano_vestibular = b.ano;
```

Deste modo, finalizamos a implementação do ambiente dimensional do DM Vestibular. No entanto, a etapa de transformação não está completa. Falta integrar os ambientes relacionais e dimensionais dos DM Graduação e DM Vestibular, para formarem ambientes únicos do ADW.

Iniciando pela integração dos modelos dimensionais, como prevê Kimball, devemos realizar as transformações que possibilitem a integração de acordo com o seguinte modelo:

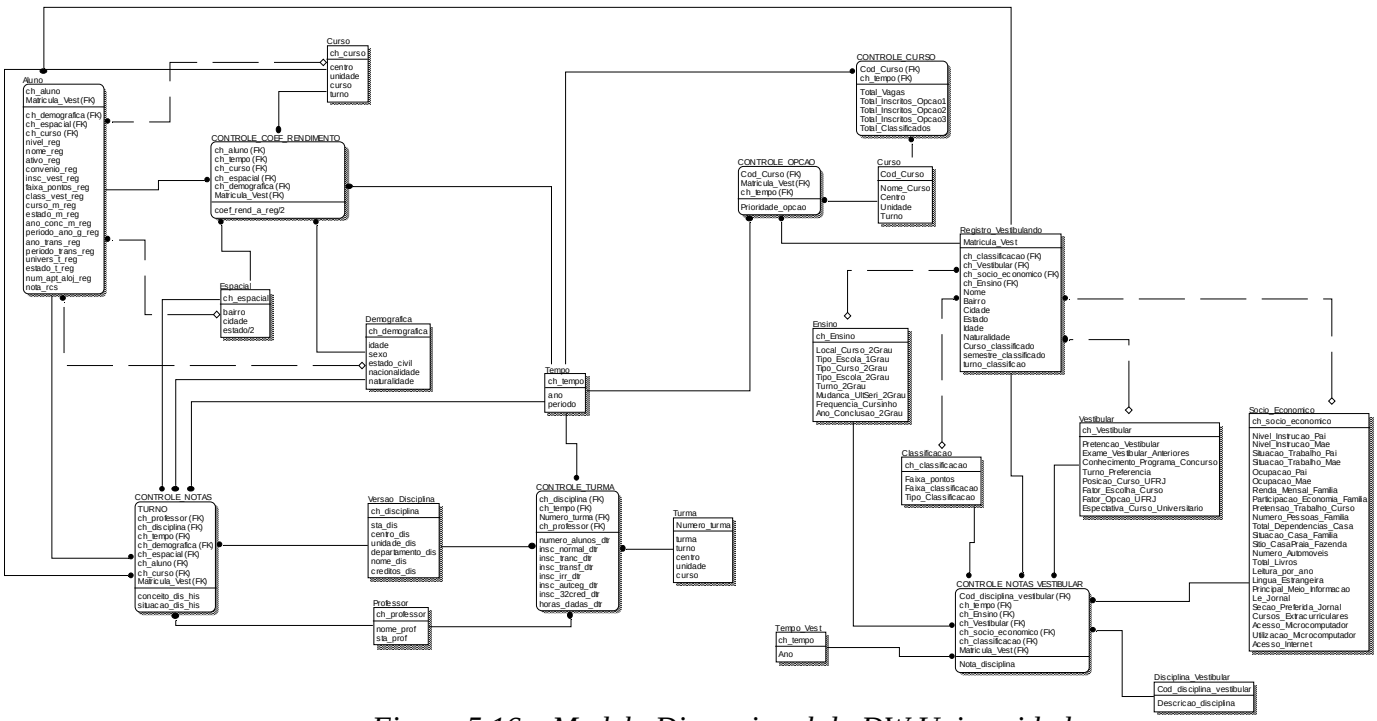

Figura 5.16 – Modelo Dimensional do DW Universidade

Ao integrar os ambientes dimensionais dos dois DM, verificamos que as tabelas de dimensões e de fatos **CONTROLE_CURSO**, **CURSO**, **CONTROLE_OPCAO** podem ser inseridas sem problemas. Da mesma forma, todas as minidimensões de **REGISTRO_VESTIBULANDO** também podem ser incluídas sem que ocorra nenhum problema de conflito ou redundância.

Notamos a presença das seguintes dimensões semelhantes: **TEMPO_CURSO** e **TEMPO** e **DISCIPLINA_VESTIBULAR** do DM Vestibular e a dimensão homônima contida no DM Graduação**.** No segundo caso, observamos que as informações que contêm são as mesmas, embora os nomes dos atributos sejam diferentes. Em ambos os casos, estas tabelas permanecerão no ambiente.

Segundo o modelo dimensional integrado do ADW, a dimensão **REGISTRO_VESTIBULANDO** tornou-se uma minidimensão da dimensão **ALUNO**, através da substituição do atributo INSC_VEST_REG na dimensão **ALUNO** pela chave dessa dimensão.

Para finalizar a integração dos dois ambientes dimensionais, o último passo a ser realizado é a exclusão da tabela de fatos **CONTROLE_VESTIBULAR** presente no DW. Isto ocorre porque esta tabela é equivalente à tabela **CONTROLE_NOTAS_VESTIBULAR** do DM Vestibular, porém esta considera todos os vestibulandos que realizaram o concurso. Desta forma, as minidimensões relacionadas a essa tabela de fatos permitem uma maior combinação de consultas. A consulta a essas informações pela dimensão **ALUNO** é obtida através da chave *MATRICULA_VEST*.

Para realizar a integração dos ambientes relacionais, como Inmon propõe, as transformações devem se basear no modelo relacional integrado do ADW, como representado abaixo:

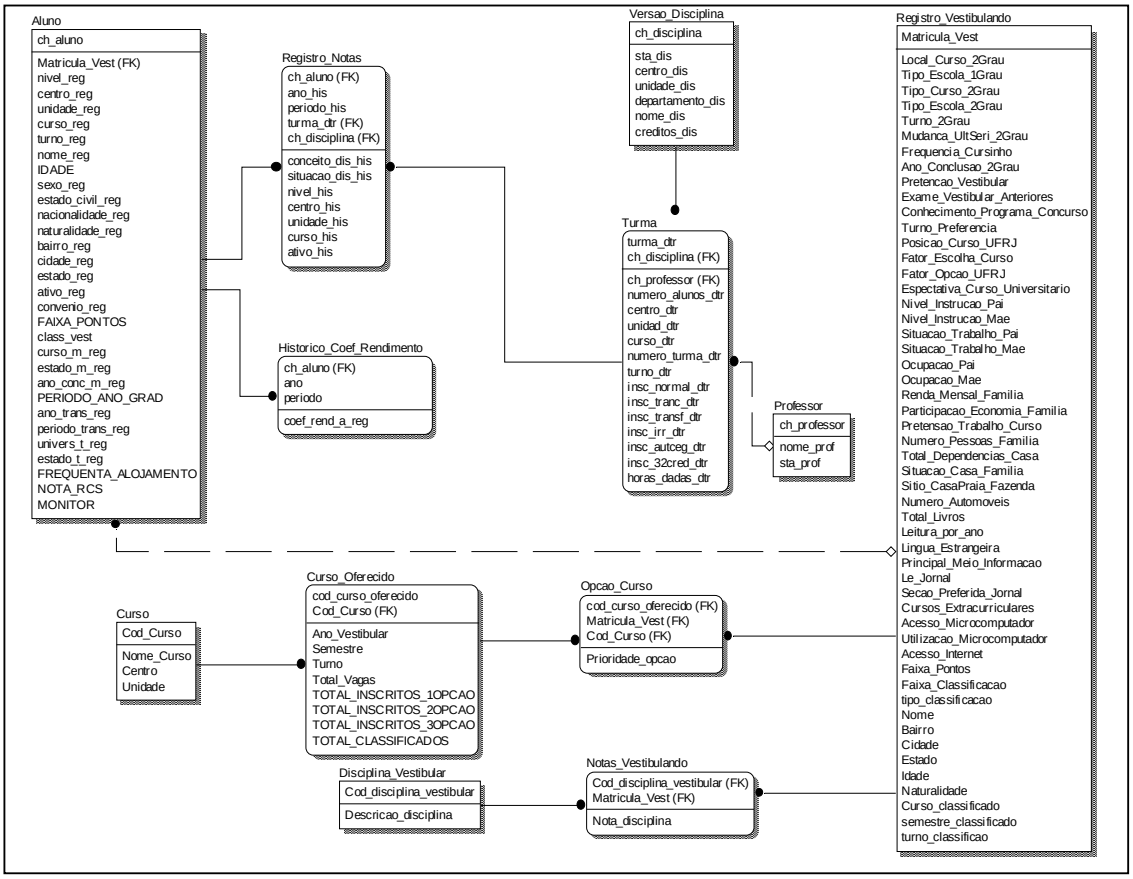

Figura 5.17 – Modelo Relacional do DW Universidade

Integrando o ambiente relacional do DM Vestibular ao ambiente relacional do DW representado pelo DM Graduação, observamos que foi possível inserir sem conflitos as tabelas *Opcao_Curso*, *Curso_Oferecido*, *Curso* e *Registro_Vestibulando.*

A tabela *Registro_Vestibulando* possui uma relação já existente com a entidade *Aluno*, através do atributo *INSC_VEST_REG* da tabela *Aluno* que contém a matrícula dos alunos no vestibular.

Devido ao fato de ser conflitante, a tabela *Disciplina_Vestibular* contida anteriormente no DW foi substituída pela tabela homônima do DM Vestibular. Com essa substituição, para se obter as notas das disciplinas no vestibular dos alunos da universidade é necessário acessar a tabela *Registro_Vestibulando* e, a partir desta, acessar a tabela *Notas_Vestibulando*. A tabela *Notas_Vestibulando* do DM Vestibular contém as informações referentes às notas dos vestibulandos por disciplina. Consequentemente, esta tabela conterá as notas dos alunos.

A integração dos dois ambientes possibilita a realização de consultas a nível de DW como, por exemplo, analisar o desempenho dos alunos com bom desempenho no vestibular de acordo com a situação socio-econômica ou pretensões para o vestibular. Estas novas consultas podem ser sugeridas aos usuários finais do DM graduação como uma nova versão.

5.4.3 Carga

A carga dos dados constitui a última etapa do processo de povoamento do ambiente de data warehouse. Na maior parte dos casos, ela é realizada em conjunto com o tratamento dos dados, através do uso dos mesmos comandos SQL. Porém, existem tabelas que, pela natureza das transformações realizadas, necessitam que os dados nela contidos sejam copiados para a tabela do ambiente de produção, de modo a finalizar todo o processo. Em geral, estas tabelas são as que requerem mais de um passo de transformação ou que necessitam de estruturas auxiliares para que as transformações possam ser realizadas.

Em relação à janela de tempo utilizada para os processos ETL, assim como a frequência das atualizações, estas não causam motivo de preocupação para o DW Universidade. Isto se deve ao fato dos processos só serem realizados a cada semestre. É claro que a primeira carga será mais demorada, pois todos os dados correspondentes aos últimos cinco anos deverão ser carregados.

Outra questão para qual devemos chamar a atenção é o fato de que provavelmente quando o sistema Graduação estiver disponível para a atualização no início do ano, o sistema Vestibular não estará. Isto não causa nenhum descompasso entre as versões dos dados, pois as informações relativas ao vestibular dos alunos cujos dados de graduação

estarão sendo carregados já foram inseridas no ambiente durante a última carga do DM Vestibular.

É nesta etapa que todos os índices e agregados devem ser criados, quando ocorrerá o planejamento e a implementação do ambiente de acesso do usuário final aos dados contidos no ADW.

Para finalizar esta etapa, basta assegurar o tratamento dos dados que por ventura foram rejeitados durante o processamento inicial, além de realizar consultas sobre as tabelas do ambiente de produção de forma a assegurar a qualidade dos dados.

5.5 Aspectos do Esquema Físico

Com base nos conceitos e orientações apresentados no capítulo 4, analisaremos alguns aspectos relativos ao projeto físico para o ADW sob nosso estudo, como as atividades de indexação e dimensionamento inicial do banco de dados.

É importante ressaltar que a análise que aqui será apresentada não é uma solução única, definitiva e nem a melhor, visto que determinadas questões de um projeto físico, como a criação de índices, devem ser construídas visando à melhoria de desempenho do ambiente de data warehouse, e isto só é possível tendo-se em mente as principais respostas que os usuários buscam quando fazem acesso aos dados.

5.5.1 Criação de Índices

Nos sistemas gerenciadores de banco de dados atuais, todas as chaves e/ou índices, como padrão, são gerados com a estrutura *B-Tree* ou similar. Sendo assim, nosso objetivo é identificar chaves e índices que possam ser criados com estruturas diferentes, de modo que se otimize o desempenho dos acessos em relação ao tipo de índice normalmente utilizado (*B-Tree*).

As chaves das tabelas podem ser criadas normalmente com estrutura *B-Tree*, pois apresentam padrões de cardinalidade adequados para este tipo.

No entanto, podemos observar que estas chaves são formadas por atributos que podem ter seus valores repetidos para mais de uma entrada nas tabelas de fatos, como é o caso de *CH_ALUNO, CH_CURSO, CH_ESPACIAL, CH_DEMOGRAFICA* e

MATRICULA_VEST na tabela de fatos **CONTROLE_COEF_RENDIMENTO** e *CH_ENSINO, CH_VESTIBULAR, CH_SOCIO_ECONOMICO, CH_CLASSIFICACAO* e *MATRICULA_VEST* na tabela de fatos **CONTROLE_NOTAS_VESTIBULAR**. Em ambos os casos, cada entrada na dimensão **ALUNO** corresponderá a uma média de 10 entradas nestas tabelas de fatos.

Sendo assim, teremos em cada uma destas tabelas, 10 linhas com mesmo valores para os campos *CH_ALUNO, CH_CURSO, CH_ESPACIAL, CH_DEMOGRAFICA* e *MATRICULA_VEST;* e *CH_ENSINO, CH_VESTIBULAR, CH_SOCIO_ECONOMICO, CH_CLASSIFICACAO* e *MATRICULA_VEST,* respectivamente. Sob o ponto de vista das chaves, a única diferença entre estas linhas estará nos valores dos atributos *CH_TEMPO* e, talvez, *COEF_REND_A_REG*, quando o coeficiente de rendimento do aluno for alterado para a tabela de fatos **CONTROLE_COEF_RENDIMENTO**; e ainda, *CH_TEMPO* e *COD_DISCIPLINA_VESTIBULAR* nas linhas presentes na tabela de fatos **CONTROLE_NOTAS_VESTIBULAR** para um mesmo aluno.

Em determinadas situações de consulta, pode ser desejável agrupar as linhas destas tabelas para cada aluno, de forma a acompanhar sua evolução a cada período da graduação, no primeiro caso, ou analisar a média de suas notas no vestibular, no segundo caso, por exemplo.

A chave *B-Tree* que foi criada sobre estas tabelas pode não ser muito eficiente para estas situações exemplificadas. Para estes casos, seria melhor criarmos um índice *Bitmap*, já que podemos considerar que um aluno gerando 10 entradas em uma tabela, a torna uma tabela com baixa cardinalidade.

Por outro lado, se analisarmos do ponto de vista de TODOS os atributos que compõem as chaves destas tabelas, não faria sentido criarmos índices *Bitmap* sobre elas, já que os atributos *CH_TEMPO* e *COD_DISCIPLINA_VESTIBULAR* estarão variando para cada entrada. Contudo, podemos nos valer do que foi dito anteriormente: SOMENTE estes atributos variam para cada entrada nestas tabelas, além, é claro, dos valores dos fatos propriamente ditos – *COEF_REND_A_REG* e *NOTA_DISCIPLINA* – mas que não fazem parte das chaves.

Desta forma, se criarmos índices *Bitmap* para estas tabelas, utilizando somente os atributos que se repetem nas chaves, estaremos atendendo de forma mais eficiente às

consultas exemplificadas anteriormente. Logo, estaríamos criando um índice *Bitmap* para **CONTROLE_COEF_RENDIMENTO** com todos os atributos que compõem sua chave, EXCETO o atributo *CH_TEMPO*, e um índice *Bitmap* para **CONTROLE_NOTAS_VESTIBULAR** com todos os atributos que compõem sua chave, EXCETO os atributos *CH_TEMPO* e *COD_DISCIPLINA_VESTIBULAR*.

A tabela de fatos **CONTROLE_NOTAS** terá no mínimo 25 entradas para cada linha da dimensão **ALUNO**. Analisando esta tabela de modo análogo às tabelas anteriores, podemos observar que um índice *Bitmap* com os atributos de sua chave, excluindo o atributo *CH_PROFESSOR*, seria o mais indicado.

Seguindo esta linha de raciocínio, podemos montar a seguinte tabela:

Nome da Tabela de Fatos	Atributos da chave não utilizados na criação de índices Bitmap com o conjunto de atributos da chave.
Controle_Coef_Rendimento	Ch_tempo
Controle_Notas_Vestibular	Ch_tempo e Cod_disciplina_vestibular
Controle_Notas	Ch_tempo e Ch_Professor
Controle_Turma	Ch_tempo e Ch_Professor
Controle_Curso	Ch_Tempo
Controle_Opcao	Ch_Tempo e Matricula_Vest

Tabela 5.5 – Lista dos atributos excluídos para a criação dos índices Bitmap

Abaixo, podemos verificar a lista de índices *Bitmap* criados para as tabelas de fatos:

Tabela sobre a qual será criado índice Bitmap	Campos utilizados na criação do índice Bitmap
Controle_Coef_Rendimento	ch_curso, ch_aluno, ch_espacial, ch_demografica
Controle_Turma	ch_disciplina, numero_turma
Controle_Notas	ch_disciplina, ch_disciplina, ch_demografica, ch_espacial, ch_aluno, ch_curso, turno
Controle_Notas_Vestibular	ch_ensino, ch_vestibular, ch_socio_economico, ch_classificacao, matricula_vest
Controle_Curso	cod_curso, total_vagas
Controle_Opcao	cod_curso, prioridade_opcao

Tabela 5.6 – Lista dos índices Bitmap criados para as tabelas de fatos

Ainda procurando por índices *Bitmap*, observamos que as categorias *FAIXA_PONTOS_REG* e *CLASS_VEST_REG* da dimensão **ALUNO** poderiam possuir um índice deste tipo, de forma a facilitar a contagem de alunos para cada faixa, por exemplo.

Enquanto que as tabelas de fatos possuem um ou no máximo dois índices grandes construídos por meio de combinação de chaves de dimensão, uma tabela de dimensão, por outro lado, pode conter vários índices, pois cada atributo textual pode ser utilizado como base para uma restrição. Desta forma, podem ser construídos índices em muitos, se não em todos os atributos de cada tabela de dimensão (KIMBALL, 1998). Para os atributos de baixa cardinalidade, recomenda-se o uso de índices *Bitmap.*

O grande número de índices em uma tabela de dimensão levanta algumas discussões administrativas. Em muitos casos, o espaço total em disco necessário para uma tabela de dimensão será o dobro do espaço necessário para os dados em si. Adicionar ou atualizar registros em tabelas de dimensões tende a ser um processo extremamente lento, principalmente se comparado a inserções de registros em tabelas de fatos. Por esta razão, vale lembrar que, durante as cargas do ADW, é importante que os índices estejam desabilitados, de modo que o processo possa ocorrer da forma mais eficiente.

5.5.2 *Dimensionamento de Banco de Dados*

Para realizarmos uma estimativa do espaço em disco que será necessário para acomodar a carga inicial do ambiente em estudo, calculamos o tamanho médio em bytes das linhas de cada tabela de dimensões e de fatos, como podemos verificar nas tabelas abaixo:

DM Graduação	
Aluno	206
Controle_Coef_Rendimento	23
Controle_Disciplina	47
Controle_Notas	54
Controle_Vestibular	29
Curso	120
Demográfica	49
Disciplina_Vestibular	48
Espacial	44
Professor	50
Tempo	17
Turma	64
Versao_Disciplina	93

DM Vestibular	
Classificação	94
Controle_Curso	29
Controle_Notas_Vestibular	42
Controle_Opcao	27
Ensino	154
Registro_Vestibulando	141
Socio_Economico	614
Vestibular	234

Tabelas 5.7 e 5.8 – Estimativas de Tamanho de Linhas para as Tabelas do ADW

O segundo passo do dimensionamento é estimar o número de linhas que cada tabela do ADW Universidade irá conter. Sendo assim, estimamos um total de 65.000 registros de alunos para o período que engloba a carga inicial do ambiente, o que faz com que a dimensão **ALUNO** ocupe inicialmente 13 MB de espaço em disco.

Com base no total de alunos considerado, verificamos que a tabela de fatos **CONTROLE_COEF_RENDIMENTO** ocupará aproximadamente 3 MB para cada ano. Assumindo que estaremos realizando a carga com dados históricos de um período de 5 anos, o tamanho inicial desta tabela será de 15 MB.

As dimensões **ESPACIAL** e **DEMOGRAFICA** devem ter aumento próximo a zero ao longo do tempo, mantendo-se com o tamanho aproximado ao da carga inicial - que estimamos em 2 MB para cada uma delas – considerando que os 65.000 alunos podem gerar aproximadamente 35.000 entradas em cada uma delas. Desta forma, elas totalizarão 4 MB.

Assumindo que a relação candidato/vaga nos vestibulares da UFRJ tem se mantido na média de 15 candidatos para uma vaga, os 65.000 alunos podem representar 975.000 entradas na dimensão **REGISTRO_VESTIBULANDO**, ao longo do tempo. Assim, 130 MB serão necessários para o armazenamento desta dimensão.

Estimamos que as 975.000 entradas na dimensão **REGISTRO_VESTIBULANDO** irão gerar uma média de 500.000 registros em cada uma das dimensões **ENSINO**, **CLASSIFICACAO**, **SOCIO_ECONOMICO** e **VESTIBULAR**, que juntas ocuparão aproximadamente 600 MB.

Com raciocínio semelhante para as demais dimensões e tabelas de fatos, chegamos à conclusão de que, para um período de histórico de 5 anos, elas necessitarão de aproximadamente 1 GB de espaço em disco. Com isto, somente a estrutura dimensional do ADW da UFRJ ocupará quase 2 GB.

De acordo com as diretivas apresentadas no capítulo 4, devemos reservar o mesmo espaço ocupado pelo ADW para os índices que possam ser gerados. Logo, necessitamos de 4 GB para a carga inicial do modelo dimensional.

Para o dimensionamento do espaço ocupado pela estrutura relacional, podemos nos basear na seguinte idéia: a grosso modo, a estrutura dimensional foi derivada da estrutura relacional através de "desmembramentos" de tabelas, para a formação das

dimensões **ESPACIAL**, **DEMOGRAFICA**, **ENSINO**, **CLASSIFICAÇÃO**, **SOCIO_ECONOMICA** e **VESTIBULAR**. De acordo com os cálculos realizados anteriormente, prevemos uma redução de aproximadamente 60% do volume de dados, ao efetuarmos estes "desmembramentos", pois estaríamos eliminando as repetições. Por exemplo, enquanto a tabela **ALUNO** possui 65.000 registros de alunos da universidade, a dimensão **ESPACIAL** possui 35.000 entradas. No entanto, na estrutura relacional, estas dimensões não existem e seus dados estão nas tabelas que a geraram, apresentando assim repetições. Logo, o espaço ocupado pela estrutura relacional será o espaço ocupado pela dimensional, acrescido do volume reduzido estimado para cada dimensão derivada.

Desta forma, a estrutura relacional será:

- 3,5 MB maior para os dados que deveriam estar nas dimensões **ESPACIAL** e **DEMOGRÁFICA** juntas, porém estão na tabela **ALUNO**;
- 1,2 GB maior para os dados que deveriam estar nas dimensões **ENSINO**, **CLASSIFICACAO**, **SOCIO_ECONOMICO** e **VESTIBULAR**, mas que estão na tabela **REGISTRO_VESTIBULANDO**.

Logo, o espaço ocupado pela estrutura relacional do ADW da UFRJ será de aproximadamente 5,5 GB, significando que para mantermos os dois ambientes - relacional e dimensional - em uma mesmo servidor, serão necessários 9,5 GB de espaço. Além disso, no servidor onde serão executados os processos de povoamento, o espaço necessário será o mesmo para abrigar as duas estruturas – visto que estamos criando e povoando estas estruturas primeiramente neste ambiente – mais o espaço adicional de cerca de 300 MB para as tabelas intermediárias.

Neste momento, não estamos considerando os espaços necessários para o funcionamento dos sistemas gerenciadores de banco de dados, que podem variar de fornecedor para fornecedor, mas que em geral, são os espaços utilizados para ordenação, criação de índices, tabelas temporárias, entre outros, e que podem chegar a 2 GB para um processamento deste porte.

CAPÍTULO 6

CONCLUSÃO

6.1 Considerações Gerais

A tecnologia de data warehousing já não pode ser considerada tão nova, apesar de seu início relativamente recente. Muitas pesquisas e projetos já foram realizados, no meio acadêmico e no mercado, para que seja obtido o máximo de proveito dela. Apesar disto, observa-se que há muitas dúvidas em relação às melhores práticas de construção do ambiente computacional que irá suportar as decisões estratégicas de uma empresa. Vários estudos já foram, e ainda serão, realizados no sentido de aprimorar cada vez mais as técnicas de desenvolvimento de um ADW, como é o caso da tese de mestrado de Vânia Soares, na qual nos baseamos para desenvolvermos este trabalho (SOARES, 1998).

Em linhas gerais, nosso trabalho visou à utilização do modelo proposto por Vânia Soares para o ADW da UFRJ, de forma que pudéssemos demonstrar como as etapas de povoamento podem ser realizadas, sendo fundamentais para o sucesso de qualquer projeto de implementação de um ambiente de data warehouse. Verificamos também que estas etapas são extremamente dependentes do modo como a modelagem foi realizada. Se esta for mal feita, um alto nível de retrabalho será necessário, principalmente no que tange à identificação dos dados presentes no ambiente operacional.

Acreditamos termos colaborado com uma visão geral das particularidades de um projeto físico para um ambiente de data warehouse, visto que esta etapa é fundamental para o projeto, por garantir que o ambiente não possua limitações de evolução e desempenho. Cabe lembrar que o projeto físico deve ser continuamente revisto, utilizando as informações de monitoramento geradas como subsídios para o seu refinamento.

6.2 Sugestões e Trabalhos Futuros

De modo a dar continuidade a este trabalho, sugerimos os seguintes estudos:

A) Arquitetura de Acesso aos Dados

Após as etapas de modelagem e construção do ambiente de data warehouse propriamente dito, seria interessante que fossem abordadas questões como planejamento e construção dos mecanismos de acesso aos dados contidos neste ambiente, assim como ferramentas que maximizem o poder de informação destes dados (por exemplo, o desenvolvimento de modelos de data mining).

B) Agregados

Embora tenhamos abordado aspectos gerais do processo de criação de agregados, não aplicamos os conceitos em nosso estudo de caso, pois a criação destes depende muito da forma como os dados serão acessados, assim como o padrão destes acessos. Sendo assim, um novo trabalho poderia abordar com mais profundidade as técnicas de construção de agregados, utilizando o mesmo estudo de caso para aplicação das mesmas.

C) Ferramentas de Extração, Transformação e Carga dos Dados

Em nosso trabalho, utilizamos *scripts* SQL para realizarmos as etapas de extração, transformação e carga dos dados. Recomendamos o uso de ferramentas específicas para que, a partir dos modelos de dados dos DMs e do DW, gerem programas responsáveis pela extração dos dados. Além de tornar o processo de povoamento do ADW mais fácil, estas ferramentas, em geral, constituem importante mecanismo de gestão de metadados.

Anexo 1

DICIONÁRIO DE DADOS REFERENTE AO ESTUDO DE CASO UNIVERSIDADE

1.1 - Modelo relacional do data mart Graduação

Entidade	Atributo	Tipo	Valor Default	Significado
Aluno	ch_aluno	char(10)		**Chave da entidade**
	Nivel_reg	char(15)	"Graduacao"	**Nível do Curso: Graduação, Extensão, Aperfeiçoamento, Especialização, Mestrado, Doutroado ou Pós-doutorado**
	Centro_reg	char(15)	Nulo	**Centro do Curso**
	Unidade_reg	char(15)	Nulo	**Unidade do Curso**
	Curso_reg	char(15)	Nulo	**Curso**
	Turno_reg	char(8)		**Turno do Curso**
	Nome_reg	char(40)		**Nome do Aluno**
	IDADE	number		**Idade do Aluno**
	Sexo_reg	char(1)		**Sexo do Aluno**
	Estado_civil_reg	char(10)		**Estado civil do Aluno**
	Nacionalidade_reg	char(15)	Nulo	**Nacionalidade do Aluno**
	Naturalidade_reg	char(15)	Nulo	**Naturalidade do Aluno**
	Bairro_reg	char(18)		**Bairro de residência do Aluno**
	cidade_reg	char(20)		**Cidade de Residência do Aluno**
	Estado_reg	char(2)		**Estado de Residência do Aluno**
	Ativo_reg	char(10)		**Situação da Matrícula do Aluno**
	Convenio_reg	char(3)		**"Sim" se for aluno convêniado ou "Não", caso contrário**
	Insc_vest_reg	char(6)		**Número de Inscrição do Aluno no vestibular**
	FAIXA_PONTOS	char(30)		**Faixa de Pontos do Aluno no vestibular**

	Class_vest_reg	char(4)		**Classificação do Aluno no Vestibular**
	Curso_m_reg	char(20)	Nulo	**Tipo de curso de 2º grau do Aluno**
	Estado_m_reg	char(2)		**Estado onde o Aluno cursou o 2º grau**
	Ano_conc_m_reg	char(2)		**Ano de conclusão do 2º grau do Aluno**
	periodo_ano_grad	char(13)		**Periodo e ano de graduação do Aluno**
	ano_trans_reg	char(4)		**Ano em que o Aluno veio transferido**
	periodo_trans_reg	char(1)		**Periodo em que o Aluno Veio transferido**
	univers_t_reg	char(20)	Nulo	**Universidade da qual o Aluno Veio transferido**
	estado_t_reg	char(2)		**Estado do qual o Aluno veio transferido**
	frequenta_alojamento	char(3)	"Não"	**"Sim" se o Aluno frequenta alojamento ou "Não", caso contrário**
	nota_rcs	char(3)	Nulo	**Nota do RCS do Aluno**
	recebe_ajuda	char(3)	"Não"	**"Sim" se o Aluno recebe ajuda ou "Não", caso contrário**
	monitor	char(3)	"Não"	**"Sim" se o Aluno é monitor ou "Não", caso contrário**
Disciplina_Vestibular	ch_aluno	char(10)		**Chave de Aluno**
	nome_dist_vest_reg	char(10)		**Nome da disciplina do Vestibular**
	pont_dist_vest_reg	char(5)		**Pontos do Aluno na disciplina do Vestibular**
Registro_Notas	ch_aluno	char(10)		**Chave de Aluno**
	ano_his	char(4)		**Ano em que o Aluno cursou a disciplina**
	periodo_his	char(1)		**Periodo em que o Aluno cursou a disciplina**
	turma_dtr	char(3)		**Turma na qual o Aluno cursou a disciplina**
	ch_disciplina	char(6)		**Chave da disciplina**

	conceito_dis_his	char(3)		**Conceito obtido pelo Aluno na disciplina**
	situacao_dis_his	char(15)		**Situação do Aluno na disciplina**
	nivel_his	char(15)		**Nivel do Aluno**
	centro_his	char(15)		**Centro do Curso do Aluno**
	unidade_his	char(15)		**Unidade do Curso do Aluno**
	curso_his	char(15)		**Curso do Aluno**
	ativo_his	char(10)		**Situção: Ativa, Trancada ou Cancelada**
Historico_Coef_Rendimento	ch_aluno	char(10)		**Chave de Aluno**
	ano	char(4)		**Ano do Coeficiente de Rendimento**
	periodo	char(1)		**Período do Coeficiente de Rendimento**
	coef_rend	char(3)		**Coeficiente de Rendimento**
Versao_Disciplina	ch_disciplina	char(6)		**Chave da disciplina**
	sta_dis	char(10)		**Situação da Disciplina: Ativa ou Desativada**
	centro_dis	char(15)		**Centro que oferece a Disciplina**
	unidade_dis	char(15)		**Unidade que oferece a Disciplina**
	departamento_dis	char(15)		**Departamento que oferece a disciplina**
	nome_dis	char(30)		**Nome da disciplina**
	creditos_dis	char(2)		**Créditos da disciplina**
Turma	turma_dtr	char(3)		**Chave de turma**
	ch_disciplina	char(6)		**Chave de disciplina**
	ch_professor	char(10)		**Chave de professor**
	numero_alunos_dtr	char(3)		**Número de alunos na turma**
	centro_dtr	char(15)	Nulo	**Centro onde se localiza a turma**
	unidade_dtr	char(15)	Nulo	**Unidade onde se localiza a turma**
	curso_dtr	char(15)	Nulo	**Curso ao qual pertence a turma**
	numero_turma_dtr	char(3)		**Número da turma**
	turno_dtr	char(8)		**Turno da turma**

	insc_normal_dtr	number		**Número de alunos com inscrição normal**
	insc_tranc_dtr	number		**Número de alunos com inscrição normal**
	insc_transf_dtr	number		**Número de alunos transferidos**
	insc_irr_dtr	number		**Número de alunos com inscrição irregular**
	insc_autceg_dtr	number		**Número de alunos com inscrição autorizada pelo CEG**
	insc_32cred_dtr	number		**Insc. mais de 32 cred.**
	horas_dadas_dtr	char(3)		**Horas de aula dadas na turma**
	Ano	char(4)		**Ano da turma**
	Periodo	char(1)		**Período da turma**
Professor	ch_professor	char(10)		**Chave de professor**
	nome_prof	char(30)		**Nome do professor**
	sta_prof	char(10)	"Ativa"	**Situação de matrícula do Professor: Ativa, Trancada ou Cancelada**

1.2 - Modelo relacional do data mart Vestibular

Entidade	Atributo	Tipo	Significado
Registro_Vestibulando	matricula_vest	char(10)	**Matricula do vestibulando**
	local_curso_1grau	char(20)	**Local do curso de 1º grau do vestibulando**
	tipo_escola_1grau	char(20)	**Tipo de escola de 1º grau do vestibulando**
	tipo_curso_2grau	char(20)	**Tipo de curso de 2º grau do vestibulando**
	tipo_escola_2grau	char(20)	**Tipo de escola de 2º grau do vestibulando**
	turno_2grau	char(10)	**Turno do 2º grau do vestibulando**
	mudanca_ultser_2grau	char(20)	**Mudança na última série do 2º grau**
	frequencia_cursinho	char(20)	**Frequencia de cursino pré-vestibular**
	ano_conclusao_2grau	char(20)	**Ano de conclusão do 2º grau**

pretensao_vestibular	char(20)	**Pretensão no vestibular**
exame_vestibulares_anteriores	char(30)	**Participação em vestibulares anteriores**
conhecimento_programa_concurso	char(20)	**Conhecimento do programa do concurso vestibular**
turno_preferencia	char(20)	**Turno de preferência para estudo**
posicao_curso_UFRJ	char(40)	**Posicao no curso**
fator_escolha_curso	char(40)	**Fator de escolha do curso**
fator_opcao_UFRJ	char(30)	**Fator de opção pela UFRJ**
expectativa_curso_universitario	char(30)	**Expectativa em relação ao curso universitário**
nivel_instrucao_pai	char(30)	**Nível de instrução do pai**
nivel_instrucao_mae	char(30)	**Nível de instrução da mãe**
situacao_trabalho_pai	char(30)	**Situação de trabalho do pai**
situacao_trabalho_mae	char(30)	**Situação de trabalho da mãe**
ocupacao_pai	char(30)	**Ocupação do pai**
ocupacao_mae	char(30)	**Ocupação da mãe**
renda_mensal_familia	char(30)	**Renda mensal da família**
participacao_economia_familia	char(30)	**Participação na economia familiar**
pretensao_trabalho_curso	char(20)	**Pretensão de trabalho enquanto cursa a universidade**
numero_pessoas_familia	char(10)	**Número de pessoas na família**
total_dependencias_casa	char(10)	**Total de dependências da casa**
situacao_casa_familia	char(30)	**Sitação da casa da família**
sitio_casapraia_fazenda	char(10)	**Propriedade de sítio, casa de praia ou fazenda**
numero_automoveis	char(20)	**Número de automóveis da família**
total_livros	char(30)	**Total de livros em casa**
leitura_por_ano	char(30)	**Número de leituras por ano**
lingua_estrangeira	char(30)	**Conhecimento de língua estrangeira**
principal_meio_informacao	char(20)	**Principal meio de informação**
le_jornal	char(30)	**Leitura de jornal**

	secao_preferida_jornal	char(30)	**Seção preferida do jornal**
	cursos_extracurriculares	char(30)	**Cursos extracurriculares**
	acesso_microcomputador	char(20)	**Acesso a microcomputador**
	utilizacao_microcomputador	char(20)	**Utilização do microcomputador**
	acesso_internet	char(20)	**Acesso a internet**
	faixa_pontos	char(30)	**Faixa de pontos no vestibular**
	faixa_classificacao	char(30)	**Faixa de classificação no vestibular**
	tipo_classificacao	char(30)	**Tipo de classificação no vestibular**
	nome	char(40)	**Nome do vestibulando**
	bairro	char(15)	**Bairro de residência do vestibulando**
	cidade	char(15)	**Ciadade de residência do vestibulando**
	estado	char(02)	**Estado de residência do vestibulando**
	idade	number	**Idade do vestibulando**
	naturalidade	char(15)	**Naturalidade do vestibulando**
	curso_classificado	char(15)	**Curso para o qual foi classificado**
	semestre_classificado	char(01)	**Semestre para o qual foi classificado**
	turno_classificado	char(08)	**Turno para o qual foi classificado**
Curso_Oferecido	cod_curso_oferecido	char(10)	**Código do curso oferecido**
	cod_curso	char(05)	**Código do curso**
	ano_vestibular	char(04)	**Ano do vestibular**
	turno	char(08)	**Turno do curso**
	total_vagas	number	**Total de vagas para o curso**
	TOTAL_INSCRITOS_1OPCAO	number	**Total de vestibulandos inscritos como 1ª opção**
	TOTAL_INSCRITOS_2OPCAO	number	**Total de vestibulandos inscritos como 2ª opção**
	TOTAL_INSCRITOS_3OPCAO	number	**Total de vestibulandos inscritos como 3ª opção**
	TOTAL_CLASSIFICADOS	number	**Total de vestibulandos classificados para o curso**
Curso	cod_curso	char(05)	**Código do curso**
	nome_curso	char(20)	**Nome do curso**
	centro	char(15)	**Centro do curso**
	unidade	char(15)	**Unidade do curso**

Opcao_Curso	cod_curso_oferecido	char(05)	**Código do curso oferecido**
	matricula_vest	char(10)	**Matricula do vestibulando**
	prioridade_opcao	char(08)	**Prioridade de opção pelo curso**
Notas_Vestibulando	cod_disciplina_vestibular	char(08)	**Código da disciplina do vestibular**
	matricula_vest	char(10)	**Matricula do vestibulando**
	nota_disciplina	number	**Nota na disciplina**
Disciplina_Vestibular	cod_disciplina_vestibular	char(08)	**Código da disciplina do vestibular**
	descricao_vestibular	char(20)	**Descrição da disciplina do vestibular**

1.3 - Modelo dimensional do data mart Graduação

Dimensão	Atributo	Tipo	Valor default	Significado
Aluno	ch_aluno	char(10)		**Chave de Aluno**
	ch_demografica	number		**Chave demográfica**
	ch_espacial	number		**Chave espacial**
	ch_curso	number		**Chave de curso**
	nivel_reg	char(15)		**Nível do Curso: Graduação, Extensão, Aperfeiçoamento, Especialização, Mestrado, Doutroado ou Pós-doutorado**
	nome_reg	char(40)		**Nome do Aluno**
	ativo_reg	char(10)		**Situação da Matrícula do Aluno**
	convenio_reg	char(3)		**"Sim" se for aluno convêniado ou "Não", caso contrário**
	insc_vest_reg	char(6)		**Número de Inscrição do Aluno no vestibular**
	faixa_pontos	char(30)		**Faixa de Pontos do Aluno no vestibular**
	class_vest_reg	char(4)		**Classificação do Aluno no Vestibular**
	curso_m_reg	char(20)	Nulo	**Tipo de curso de 2º grau do Aluno**

	estado_m_reg	char(2)		**Estado onde o Aluno cursou o 2º grau**
	ano_conc_m_reg	char(2)		**Ano de conclusão do 2º grau do Aluno**
	periodo_ano_grad	char(13)		**Periodo e ano de graduação do Aluno**
	ano_trans_reg	char(4)		**Ano em que o Aluno veio transferido**
	periodo_trans_reg	char(1)		**Periodo em que o Aluno Veio transferido**
	univers_t_reg	char(20)	Nulo	**Universidade da qual o Aluno Veio transferido**
	estado_t_reg	char(2)		**Estado do qual o Aluno veio transferido**
	frequenta_alojamento	char(3)	"Não"	**"Sim" se o Aluno frequenta alojamento ou "Não", caso contrário**
	nota_rcs	char(3)	Nulo	**Nota do RCS do Aluno**
	recebe_ajuda	char(3)	"Não"	**"Sim" se o Aluno recebe ajuda ou "Não", caso contrário**
	monitor	char(3)	"Não"	**"Sim" se o Aluno é monitor ou "Não", caso contrário**
Disciplina_Vestibular	ch_disciplina_vestibular	char(10)		**Chave da disciplina do Vestibular**
	nome_dist_vest_reg	char(10)		**Nome da disciplina do Vestibular**
Curso	ch_curso	number		**Chave de Curso**
	centro	char(15)		**Centro do Curso**
	unidade	char(15)		**Unidade do Curso**
	curso	char(15)		**Nome do Curso**
	turno	char(8)		**Turno do Curso**
Espacial	ch_espacial	number		**Chave da dimensão espacial**
	bairro	char(18)		**Bairro da dimensão**
	cidade	char(20)		**Cidade da dimensão**
	estado	char(2)		**Estado da dimensão**

Demográfica	ch_demografica	number		**Chave da dimensão demográfica**
	idade	number		**Idade da dimensão**
	sexo	char(1)		**Sexo da dimensão**
	estado_civil	char(10)		**Estado civil da dimensão**
	nacionalidade	char(15)		**Nacionalidade da dimensão**
	naturalidade	char(15)		**Naturalidade da dimensão**
Professor	ch_professor	char(10)		**Chave de professor**
	nome_prof	char(30)		**Nome do professor**
	sta_prof	char(10)		**Situação de matrícula do Professor: Ativa, Trancada ou Cancelada**
Tempo	ch_tempo	number		**Chave da dimensão Tempo**
	ano	char(4)		**Ano da dimensão**
	periodo	char(1)		**Período da dimensão**
Versao_Disciplina	ch_disciplina	char(6)		**Chave da dimensão Versão_Disciplina**
	sta_dis	char(10)		**Situação da Disciplina: Ativa ou Desativada**
	centro_dis	char(15)		**Centro que oferece a Disciplina**
	unidade_dis	char(15)		**Unidade que oferece a Disciplina**
	departamento_dis	char(15)		**Departamento que oferece a disciplina**
	nome_dis	char(30)		**Nome da disciplina**
	creditos_dis	char(2)		**Créditos da disciplina**
Turma	numero_turma	char(3)		**Número da turma**
	turma	char(3)		**Chave de turma**
	turno	char(8)		**Turno da turma**
	centro	char(15)		**Centro onde se localiza a turma**
	unidade	char(15)		**Unidade onde se localiza a turma**
	curso	char(15)		**Curso ao qual pertence a turma**

Tabela de fatos	**Atributo**	**Tipo**	**Significado**
Controle_Coef_Rendimento	ch_curso	number	**Chave de curso**

	ch_aluno	number	**Chave de Aluno**
	ch_espacial	number	**Chave da dimensão espacial**
	ch_demografica	number	**Chave da dimensão demográfica**
	ch_tempo	number	**Chave da dimensão Tempo**
	coef_rend_a_reg	char(3)	**Coeficiente de rendimento**
Controle_Vestibular	ch_aluno	number	**Chave de Aluno**
	ch_curso	number	**Chave de curso**
	ch_espacial	number	**Chave da dimensão espacial**
	ch_demografica	number	**Chave da dimensão demográfica**
	ch_disciplina_vestibular	number	**Chave da dimensão Disciplina_Vestibular**
	ch_tempo	number	**Chave da dimensão Tempo**
	pont_dis_vest_reg	char(5)	**Pontos na disciplina do vestibular**
Controle_Disciplina	ch_disciplina	number	**Chave da disciplina**
	ch_tempo	number	**Chave da dimensão Tempo**
	numero_turma	char(3)	**Numero da turma**
	ch_professor	number	**Chave de professor**
	numero_alunos_dtr	number	**Número de alunos na turma**
	insc_normal_dtr	number	**Número de alunos com inscrição normal**
	insc_tranc_dtr	number	**Número de alunos com inscrição normal**
	insc_transf_dtr	number	**Número de alunos transferidos**
	insc_irr_dtr	number	**Número de alunos com inscrição irregular**
	insc_autceg_dtr	number	**Número de alunos com inscrição autorizada pelo CEG**
	insc_32cred_dtr	number	**Insc. mais de 32 cred.**
	horas_dadas_dtr	number	**Horas de aula dadas na turma**
Controle_Notas	ch_professor	number	**Chave de professor**
	ch_disciplina	number	**Chave de disciplina**
	ch_tempo	number	**Chave da dimensão Tempo**
	ch_demografica	number	**Chave da dimensão demográfica**
	ch_espacial	number	**Chave da dimensão espacial**
	ch_aluno	number	**Chave de Aluno**
	ch_curso	number	**Chave de curso**
	turno	char(8)	**Turno**
	conceito_dis_his	char(3)	**Conceito obtido**
	situacao_dis_his	char(15)	**Situação da inscrição**

1.4 - Modelo dimensional do data mart Vestibular

Dimensão	Atributo	Tipo	Significado
Registro_Vestibulando	matricula_vest	char(10)	**Matricula do vestibulando**
	ch_classificacao	number	**Chave da dimensão Classificação**
	ch_vestibular	number	**Chave da dimensão Vestibular**
	ch_socio_economico	number	**Chave da dimensão Socio_Economico**
	ch_ensino	number	**Chave da dimensão Ensino**
	nome	char(40)	**Nome do vestibulando**
	bairro	char(15)	**Bairro de residência do vestibulando**
	cidade	char(15)	**Cidade de residência do vestibulando**
	estado	char(02)	**Estado de residência do vestibulando**
	idade	number	**Idade do vestibulando**
	naturalidade	char(15)	**Naturalidade do vestibulando**
	curso_classificado	char(15)	**Curso para o qual o vestibulando foi classificado**
	semestre_classificado	char(01)	**Semestre para o qual o vestibulando foi classificado**
	turno_classificado	char(08)	**Turno para o qual o vestibulando foi classificado**
Classificação	ch_classificacao	number	**Chave da dimensão Classificação**
	faixa_pontos	char(30)	**Faixa de pontos no vestibular**
	faixa_classificacao	char(30)	**Faixa de classificação no vestibular**
	tipo_classificacao	char(30)	**Tipo de classificação no vestibular**
Ensino	ch_ensino	number	**Chave da dimensão Ensino**
	local_curso_1Grau	char(20)	**Local do curso de 1º grau do vestibulando**
	tipo_escola_1grau	char(20)	**Tipo de escola de 1º grau do vestibulando**
	tipo_curso_2grau	char(20)	**Tipo de curso de 2º grau do vestibulando**

	tipo_escola_2grau	char(20)	**Tipo de escola de 2º grau do vestibulando**
	turno_2grau	char(10)	**Turno do 2º grau do vestibulando**
	mudanca_ultser_2grau	char(20)	**Mudança na última série do 2º grau**
	frequencia_cursinho	char(20)	**Frequencia de cursino pré-vestibular**
	ano_conclusao_2grau	char(20)	**Ano de conclusão do 2º grau**
Vestibular	ch_vestibular	number	**Chave da dimensão Vestibular**
	pretensao_vestibular	char(20)	**Pretensão no vestibular**
	exame_vestibulares_anteriores	char(30)	**Participação em vestibulares anteriores**
	conhecimento_programa_concurso	char(20)	**Conhecimento do programa do concurso vestibular**
	turno_preferencia	char(20)	**Turno de preferência para estudo**
	posicao_curso_UFRJ	char(40)	**Posição no curso**
	fator_escolha_curso	char(40)	**Fator de escolha do curso**
	fator_opcao_UFRJ	char(30)	**Fator de opção pela UFRJ**
	expectativa_curso_universitario	char(30)	**Expectativa em relação ao curso universitário**
Socio_Economico	ch_socio_economico	number	**Chave da dimensão Socio_Economico**
	nivel_instrucao_pai	char(30)	**Nível de instrução do pai**
	nivel_instrucao_mae	char(30)	**Nível de instrução da mãe**
	situacao_trabalho_pai	char(30)	**Situação de trabalho do pai**
	situacao_trabalho_mae	char(30)	**Situação de trabalho da mãe**
	ocupacao_pai	char(30)	**Ocupação do pai**
	ocupacao_mae	char(30)	**Ocupação da mãe**
	renda_mensal_familia	char(30)	**Renda mensal da família**
	participacao_economia_familia	char(30)	**Participação na economia familiar**
	pretensao_trabalho_curso	char(20)	**Pretensão de trabalho enquanto cursa a universidade**
	numero_pessoas_familia	char(10)	**Número de pessoas na família**
	total_dependencias_casa	char(10)	**Total de dependências da casa**
	situacao_casa_familia	char(30)	**Sitação da casa da família**

	sitio_casapraia_fazenda	char(10)	**Propriedade de sítio, casa de praia ou fazenda**
	numero_automoveis	char(20)	**Número de automóveis da família**
	total_livros	char(30)	**Total de livros em casa**
	leitura_por_ano	char(30)	**Número de leituras por ano**
	lingua_estrangeira	char(30)	**Conhecimento de língua estrangeira**
	principal_meio_informacao	char(20)	**Principal meio de informação**
	le_jornal	char(30)	**Leitura de jornal**
	secao_preferida_jornal	char(30)	**Seção preferida do jornal**
	cursos_extracurriculares	char(30)	**Cursos extracurriculares**
	acesso_microcomputador	char(20)	**Acesso a microcomputador**
	utilizacao_microcomputador	char(20)	**Utilização do microcomputador**
	acesso_internet	char(20)	**Acesso a internet**
Tempo	ch_tempo	number	**Chave da dimensão Tempo**
	ano	char(4)	**Ano da dimensão**
Disciplina_Vestibular	ch_disciplina_vestibular	char(08)	**Chave da dimensão Disciplina_Vestibular**
	descricao_disciplina	char(20)	**Descrição da disciplina do vestibular**
Curso	cod_curso	char(05)	**Código do Curso**
	nome_curso	char(20)	**Nome do Curso**
	turno	char(08)	**Turno do Curso**
	centro	char(15)	**Centro do Curso**
	unidade	char(15)	**Unidade do Curso**

Tabela de Fatos	**Atributo**	**Tipo**	**Significado**
Controle_Notas_Vestibular	cod_disciplina_vestibular	char(08)	**Código da disciplina do vestibular**
	ch_tempo	number	**Chave da dimensão Tempo**
	ch_ensino	number	**Chave da dimensão Ensino**
	ch_vestibular	number	**Chave da dimensão Vestibular**
	ch_socio_economico	number	**Chave da dimensão Socio_Economico**
	ch_classificacao	number	**Chave da dimensão Classificação**
	matricula_vest	char(10)	**Matricula do vestibulando**
	nota_disciplina	number	**Nota na disciplina**
Controle_Curso	cod_curso	char(05)	**Código do Curso**

	ch_tempo	number	**Chave da dimensão Tempo**
	total_vagas	number	**Total de vagas para o curso**
	TOTAL_INSCRITOS_1OPCAO	number	**Total de vestibulandos inscritos como 1ª opção**
	TOTAL_INSCRITOS_2OPCAO	number	**Total de vestibulandos inscritos como 2ª opção**
	TOTAL_INSCRITOS_3OPCAO	number	**Total de vestibulandos inscritos como 3ª opção**
	TOTAL_CLASSIFICADOS	number	**Total de vestibulandos classificados para o curso**
Controle_Opcao	cod_curso	char(05)	**Código do Curso**
	matricula_vest	char(10)	**Matricula do vestibulando**
	ch_tempo	number	**Chave da dimensão Tempo**
	prioridade_opcao	char(08)	**Prioridade de opção pelo curso**

Anexo 2

SCRIPTS DE EXTRAÇÃO, TRANSFORMAÇÃO E CARGA DOS DADOS DO ESTUDO DE CASO UNIVERSIDADE

I) DM Graduação

I .1) Script de criação e povoamento das entidades do modelo relacional

```
/*************************************************************

    CRIACAO DA ESTRUTURA RELACIONAL
    DO DATA MART  DO SISTEMA DE GRADUACAO

*************************************************************/

/*************************************************************
      ENTIDADE ALUNO
*************************************************************/


CREATE TABLE staging.aluno (
ch_aluno                char(10) PRIMARY KEY NOT NULL,
nivel_reg               char(15)    DEFAULT  'Graduacao',
centro_reg              char(15)    DEFAULT NULL,
unidade_reg             char(15)    DEFAULT NULL,
curso_reg               char(15)    DEFAULT NULL,
turno_reg               char(8),
nome_reg                char(40),
IDADE                   number,
sexo_reg                char(1) ,
estado_civil_reg        char(10),
nacionalidade_reg       char(15)    DEFAULT NULL ,
naturalidade_reg        char(15)    DEFAULT NULL,
bairro_reg              char(18),
cidade_reg              char(20),
estado_reg              char(2),
ativo_reg               char(10),
convenio_reg            char(3),
insc_vest_reg           char(6),
FAIXA_PONTOS            char(30),
class_vest_reg          char(4),
curso_m_reg             char(20)    DEFAULT NULL,
estado_m_reg            char(2),
ano_conc_m_reg          char(2),
PERIODO_ANO_GRAD        char(13),
ano_trans_reg           char(4),
periodo_trans_reg       char(1),
univers_t_reg           char(20)    DEFAULT NULL,
estado_t_reg            char(2),
FREQUENTA_ALOJAMENTO    char(3)     DEFAULT 'Nao',
```

```
NOTA_RCS                char(3)     DEFAULT NULL,
RECEBE_AJUDA            char(3)     DEFAULT 'Nao',
MONITOR                 char(3)     DEFAULT 'Nao'
);


/*************************************************************
      ENTIDADE DISCIPLINA_VESTIBULAR
*************************************************************/


CREATE TABLE staging.disciplina_vestibular (

ch_aluno                char(10),
nome_dist_vest_reg      char(10),
pont_dist_vest_reg      char(5)
);

/*************************************************************
      ENTIDADE REGISTRO_NOTAS
*************************************************************/


CREATE TABLE staging.registro_notas (

ch_aluno          char(10),
ano_his           char(4),
periodo_his       char(1),
turma_dtr         char(3),
ch_disciplina     char(6),
conceito_dis_his  char(3),
situacao_dis_his  char(15),
nivel_his         char(15),
centro_his        char(15),
unidade_his       char(15),
curso_his         char(15),
ativo_his         char(10)
);


/*************************************************************
     ENTIDADE HISTORICO_COEF_RENDIMENTO
*************************************************************/


CREATE TABLE staging.historico_coef_rendimento (

ch_aluno          char(10),
ano               char(4),
periodo           char(1),
coef_rend         char(3)
);
```

```
/*************************************************************
      ENTIDADE VERSAO_DISCIPLINA
*************************************************************/


CREATE TABLE staging.versao_disciplina (

ch_disciplina     char(6),
sta_dis           char(10),
centro_dis        char(15),
unidade_dis       char(15),
departamento_dis  char(15),
nome_dis          char(30),
creditos_dis      char(2)
);


/*************************************************************
      ENTIDADE TURMA
*************************************************************/


CREATE TABLE staging.turma (

turma_dtr         char(3),
ch_disciplina     char(6),
ch_professor      char(10),
numero_alunos_dtr char(3),
centro_dtr        char(15) DEFAULT NULL,
unidade_dtr       char(15) DEFAULT NULL,
curso_dtr         char(15) DEFAULT NULL,
numero_turma_dtr  char(3),
turno_dtr         char(8),
insc_normal_dtr   number,
insc_tranc_dtr    number,
insc_transf_dtr   number,
insc_irr_dtr      number,
insc_autceg_dtr   number,
insc_32cred_dtr   number,
horas_dadas_dtr   char(3),
ano               char(4),
periodo           char(1)
);


/*************************************************************
      ENTIDADE PROFESSOR
*************************************************************/


CREATE TABLE staging.professor (

ch_professor      char(10),
nome_prof         char(30),
sta_prof          char(10) DEFAULT 'Ativa'
);
```

```
/*****************************************************************

   POVOAMENTO  DA ESTRUTURA RELACIONAL
   DO DATA MART  DO SISTEMA DE GRADUACAO

*****************************************************************/

/*****************************************************************
      ENTIDADE ALUNO
*****************************************************************/


INSERT INTO staging.aluno
(
      ch_aluno,
      nivel_reg,
      centro_reg,
      unidade_reg,
      curso_reg,
      turno_reg,
      nome_reg,
      IDADE,
      sexo_reg,
      estado_civil_reg,
      nacionalidade_reg,
      naturalidade_reg,
      bairro_reg,
      cidade_reg,
      estado_reg,
      aitvo_reg,
      convenio_reg,
      insc_vest_reg,
      class_vest_reg,
      curso_m_reg,
      estado_m_reg,
      ano_conc_m_reg,
      PERIODO_ANO_GRAD,
      ano_trans_reg,
      periodo_trans_reg,
      univers_t_reg,
      estado_t_reg
)
SELECT
      numero_reg+"-"+dev_reg,
      nivel_reg,
      centro_reg,
      unidade_reg,
      curso_reg,
      turno_reg,
      nome_reg,
      TODAY() - TO_DATE(dia_n_reg+'/'+mes_n_reg+'/'+ano_n_reg),
      sexo_reg,
      estado_civil_reg,
      nacionalidade_reg,
      naturalidade_reg,
      bairro_reg,
      cidade_reg,
```

```
      estado_reg,
      aitvo_reg,
      convenio_reg,
      insc_vest_reg,
      class_vest_reg,
      curso_m_reg,
      estado_m_reg,
      ano_conc_m_reg,
      ano_trans_reg,
      periodo_trans_reg,
      univers_t_reg,
      estado_t_reg
FROM fonte.aluno;


UPDATE staging.aluno
SET nivel_reg = 'Graduação'
WHERE nivel_reg  in ( '1             ', '2             ' ,'3
') ;

UPDATE staging.aluno
SET nivel_reg = 'Extensão'
WHERE nivel_reg = '4             ';

UPDATE staging.aluno
SET nivel_reg = 'Aperfeiçoamento'
WHERE nivel_reg = '5             ';

UPDATE staging.aluno
SET nivel_reg = 'Especialização'
WHERE nivel_reg = '6             ';

UPDATE staging.aluno
SET nivel_reg = 'Mestrado'
WHERE nivel_reg = '7             ';

UPDATE staging.aluno
SET nivel_reg = 'Doutorado'
WHERE nivel_reg = '8             ';

UPDATE staging.aluno
SET nivel_reg = 'Pós-doutorado'
WHERE nivel_reg = '9             ';


UPDATE staging.aluno
FROM fonte.centro
SET staging.aluno.centro_reg = fonte.centro.desc_centro
WHERE staging.aluno.centro_reg = fonte.centro.cod_centro;

UPDATE staging.aluno
FROM fonte.unidade
SET staging.aluno.unidade_reg = fonte.centro.desc_unidade
WHERE staging.aluno.unidade_reg = fonte.unidade.cod_unidade;
```

```
UPDATE staging.aluno
FROM fonte.curso
SET staging.aluno.curso_reg  = fonte.curso.desc_curso
WHERE staging.aluno.curso_reg = fonte.curso.cod_curso;

UPDATE staging.aluno
SET turno_reg = 'Manhã'
WHERE turno_reg  in ( 'M       ');


UPDATE staging.aluno
SET turno_reg = 'Tarde'
WHERE turno_reg  in ( 'T       ');


UPDATE staging.aluno
SET turno_reg = 'Noite'
WHERE turno_reg  in ( 'N       ');

UPDATE staging.aluno
SET turno_reg = Integral'
WHERE turno_reg  in ( 'I       ');

UPDATE staging.aluno
SET sexo_reg = 'M'
WHERE sexo_reg = '0';

UPDATE staging.aluno
SET sexo_reg = 'F'
WHERE sexo_reg = '1';

UPDATE staging.aluno
FROM fonte.estado_civil
SET staging.aluno.estado_civil_reg = fonte.estado_civil.desc_estado_civil
WHERE staging.aluno.estado_civil_reg =
fonte.estado_civil.cod_estado_civil;

UPDATE staging.aluno
SET nacionalidade_reg = 'Brasileiro'
WHERE nacionalidade_reg = '1              ';

UPDATE staging.aluno
SET nacionalidade_reg = 'Naturalizado'
WHERE nacionalidade_reg = '2              ';

UPDATE staging.aluno
SET nacionalidade_reg = 'Estrangeiro'
WHERE nacionalidade_reg = '3              ';

UPDATE staging.aluno
FROM fonte.naturalidade
SET staging.aluno.naturalidade_reg = fonte.naturalidade.desc_naturalidade
WHERE staging.aluno.naturalidade_reg =
fonte.naturalidade.cod_naturalidade;
```

```
UPDATE staging.aluno
SET aitvo_reg = 'Ativa'
WHERE ativo_reg = 'A          ';

UPDATE staging.aluno
SET aitvo_reg = 'Trancada'
WHERE aitvo_reg = 'T          ';

UPDATE staging.aluno
SET aitvo_reg = 'Cancelada'
WHERE aitvo_reg = 'C          ';


UPDATE staging.aluno
FROM fonte.curso_medio
SET staging.aluno.curso_m_reg = fonte.curso_medio.desc_curso_medio
WHERE staging.aluno.curso_m_reg = fonte.naturalidade.cod_curso_medio;

UPDATE staging.aluno
FROM fonte.universidade
SET staging.aluno.univers_t_reg = fonte.universidade.desc_universidade
WHERE staging.aluno.univers_t_reg = fonte.universidade.cod_universidade;

UPDATE staging.aluno
FROM fonte.convenio
SET staging.aluno.convenio_reg = fonte.convenio.desc_convenio
WHERE staging.aluno.convenio_reg = fonte.convenio.cod_convenio;

/* CRIANDO CATEGORIAS */

UPDATE staging.aluno
FROM fonte.aluno
SET  faixa_pontos_vest = 'PONTUACAO INFERIOR A 5000'
WHERE       fonte.aluno.numero_reg = substr(staging.aluno.ch_aluno,1,8)
AND   fonte.aluno.dv_reg      = substr(staging.aluno.ch_aluno,10,1)
AND   fonte.pontos_vest_reg < 5000;

UPDATE staging.aluno
FROM fonte.aluno
SET  faixa_pontos_vest = 'PONTUACAO ENTRE 5000 e 7000'
WHERE       fonte.aluno.numero_reg = substr(staging.aluno.ch_aluno,1,8)
AND   fonte.aluno.dv_reg      = substr(staging.aluno.ch_aluno,10,1)
AND   fonte.pontos_vest_reg >= 5000
AND   fonte.pontos_vest_reg < 7000;

UPDATE staging.aluno
FROM fonte.aluno
SET  faixa_pontos_vest = 'PONTUACAO ENTRE 7000 e 8000'
WHERE       fonte.aluno.numero_reg = substr(staging.aluno.ch_aluno,1,8)
AND   fonte.aluno.dv_reg      = substr(staging.aluno.ch_aluno,10,1)
AND   fonte.pontos_vest_reg >= 7000
AND   fonte.pontos_vest_reg < 8000;
```

```
UPDATE staging.aluno
FROM fonte.aluno
SET  faixa_pontos_vest = 'PONTUACAO ACIMA de 8000'
WHERE       fonte.aluno.numero_reg = substr(staging.aluno.ch_aluno,1,8)
AND   fonte.aluno.dv_reg      = substr(staging.aluno.ch_aluno,10,1)
AND   fonte.pontos_vest_reg > 8000;

UPDATE staging.aluno
FROM fonte.aluno
SET periodo_ano_grad ='1PERIODO'+CHR(fonte.aluno.ano_g_reg)
WHERE       fonte.aluno.numero_reg = substr(staging.aluno.ch_aluno,1,8)
AND   fonte.aluno.dv_reg      = substr(staging.aluno.ch_aluno,10,1)
AND   fonte.aluno.ano_g_reg < 7;

UPDATE staging.aluno
FROM fonte.aluno
SET periodo_ano_grad = '2PERIODO'+CHR(fonte.aluno.ano_g_reg)
WHERE       fonte.aluno.numero_reg = substr(staging.aluno.ch_aluno,1,8)
AND   fonte.aluno.dv_reg      = substr(staging.aluno.ch_aluno,10,1)
AND   fonte.aluno.ano_g_reg >= 7;


/ * ARTEFATOS */

UPDATE staging.aluno
SET frequenta_alojamento = 'Sim'
WHERE ch_aluno in (
              SELECT numero_aloj+"-"+dv_aloj
              FROM fonte.alojamento
             );

UPDATE staging.aluno
SET recebe_ajuda = 'Sim'
WHERE ch_aluno in (
              SELECT numero_ajuda+"-"+dv_ajuda
              FROM fonte.ajuda_custo
             );

UPDATE staging.aluno
SET monitor = 'Sim'
WHERE ch_aluno in (
              SELECT numero_monitor+"-"+dv_monitor
              FROM fonte.monitor
             );

UPDATE staging.aluno
FROM fonte.aluno
SET staging.aluno.nota_rcs  = fonte.aluno.nota_rcs
WHERE       fonte.aluno.numero_rcs = substr(staging.aluno.ch_aluno,1,8)
AND   fonte.aluno.dv_rcs      = substr(staging.aluno.ch_aluno,10,1);
```

```
/*************************************************************
      ENTIDADE REGISTRO_NOTAS
*************************************************************/

INSERT INTO staging.registro_notas
SELECT
      a.numero_his+'-'+a.dv_his_g2,
      a.ano_his,
      a.periodo_his,
      b.turma_dis_his,
      b.part_alfa_dis_his+'-'+b.parte_num_dis_his,
      b.conceito_dis_his,
      b.situacao_dis_his,
      a.nivel_his,
      a.centro_his,
      a.unidade_his,
      a.curso_his,
      a.ativo_his,

FROM fonte.historico a, fonte.historico_disciplina b
WHERE       a.numero_his = b.numero_his
AND   a.dv_his = b.dv_his
AND   a.ano_his = b.ano_his;

UPDATE staging.registro_notas
FROM fonte.situacao
SET staging.registro_notas.situacao_dis_his =
fonte.situacao.desc_situacao
WHERE staging.registro_notas.situacao_dis_his =
fonte.situacao.cod_situacao;


UPDATE staging.registro_notas
FROM fonte.centro
SET staging.registro_notas.centro_his = fonte.centro.desc_centro
WHERE staging.registro_notas.centro_his = fonte.centro.cod_centro;

UPDATE staging.registro_notas
FROM fonte.unidade
SET staging.registro_notas.unidade_reg = fonte.centro.desc_unidade
WHERE staging.registro_notas.unidade_reg = fonte.unidade.cod_unidade;

UPDATE staging.registro_notas
FROM fonte.curso
SET staging.registro_notas.curso_reg  = fonte.curso.desc_curso
WHERE staging.registro_notas.curso_reg = fonte.curso.cod_curso;

UPDATE staging.registro_notas
SET aitvo_reg = 'Ativa'
WHERE ativo_reg = 'A          ';

UPDATE staging..registro_notas
SET aitvo_reg = 'Trancada'
WHERE aitvo_reg = 'T          ';
```

```
UPDATE staging..registro_notas
SET aitvo_reg = 'Cancelada'
WHERE aitvo_reg = 'C          ';

/*************************************************************
            ENTIDADE DISCIPLINA_VESTIBULAR
*************************************************************/

INSERT INTO staging.disciplina_vestibular
SELECT
      numero_reg+'-'+dv_reg,
      nome_dis_vest_reg,
      pont_dis_vest_reg

FROM fonte.disciplina_vestibular;

/*************************************************************
            ENTIDADE VERSAO_DISCIPLINA
*************************************************************/
INSERT INTO staging.versao_disciplina
SELECT
      parte_alfa_dis+parte_num_dis,
      sta_dis,
      centro_dis,
      unidade_dis,
      departamento_dis,
      nome_dis,
      creditos_dis
FROM fonte.versao_disciplina;

UPDATE staging.versao_disciplina
FROM fonte.centro
SET staging.versao_disciplina.centro_dis = fonte.centro.desc_centro
WHERE staging.versao_disciplina.centro_dis = fonte.centro.cod_centro;

UPDATE staging.versao_disciplina
FROM fonte.unidade
SET staging.versao_disciplina.unidade_dis = fonte.centro.desc_unidade
WHERE staging.versao_disciplina.unidade_dis = fonte.unidade.cod_unidade;

UPDATE staging.versao_disciplina
FROM fonte.departamento
SET staging.versao_disciplina.curso_dis  = fonte.curso.desc_depart
WHERE staging.versao_disciplina.curso_dis = fonte.curso.cod_depart;

UPDATE staging.versao_disciplina
SET sta_dis = 'Ativa'
WHERE sta_dis = 'A         ';

UPDATE staging.versao_disciplina
SET sta_dis = 'Desativada'
WHERE sta_dis = 'D         ';
```

```
/*************************************************************
            ENTIDADE PROFESSOR
*************************************************************/

INSERT INTO staging.professor
SELECT
      num_reg_prof+'-'+dv_reg_prof,
      nome_prof,
      sta_prof
FROM fonte.professor;

UPDATE staging.professor
SET sta_prof = 'Ativa'
WHERE sta_prof = 'A         ';

UPDATE staging.professor
SET sta_prof = 'Trancada'
WHERE sta_prof = 'T         ';

UPDATE staging.professor
SET sta_prof = 'Cancelada'
WHERE sta_prof = 'C         ';

/*************************************************************
            ENTIDADE TURMA
*************************************************************/

INSERT INTO staging.turma
SELECT
      a.turma_dtr,
      a.aparte_alfa_dtr+part_num_dtr,
      a.num_reg_prof_dtr+'-'+dv_reg_prof_dtr,
      a.centro_dtr,
      a.unidade_dtr,
      a.curso_dtr,
      a.numero_turma_dtr,
      a.turno_dtr,
      a.insc_normal_dtr,
      a.insc_tranc_dtr,
      a.insc_transf_dtr,
      a.insc_irr_dtr,
      a.insc_autceg_dtr,
      a.insc_32cred_dtr,
      a.horas_dadas_dtr
      b.eqv_ano,
      b.eqv_per
FROM
      fonte.turma a,
      fonte.equivalencia b
WHERE
      a.parte_alfa_dtr = b.parte_alfa_dis
AND   a.part_num_dtr   = b.parte_num_dis

UPDATE staging.turma
FROM fonte.centro
SET staging.registro_notas.centro_dtr = fonte.centro.desc_centro
WHERE staging.registro_notas.centro_dtr = fonte.centro.cod_centro;
```

```
UPDATE staging.turma
FROM fonte.unidade
SET staging.registro_notas.unidade_dtr = fonte.centro.desc_unidade
WHERE staging.registro_notas.unidade_dtr = fonte.unidade.cod_unidade;

UPDATE staging.turma
FROM fonte.curso
SET staging.registro_notas.curso_dtr  = fonte.curso.desc_curso
WHERE staging.registro_notas.curso_dtr = fonte.curso.cod_curso;

UPDATE staging.turma
SET turno_dtr = 'Manhã'
WHERE turno_reg  in ( 'M       ');


UPDATE staging.turma
SET turno_dtr = 'Tarde'
WHERE turno_dtr  in ( 'T       ');


UPDATE staging.turma
SET turno_dtr = 'Noite'
WHERE turno_dtr  in ( 'N       ');

UPDATE staging.turma
SET turno_dtr = 'Integral'
WHERE turno_dtr  in ( 'I       ');

/*************************************************************
    ENTIDADE HISTORICO_COEF_RENDIMENTO
*************************************************************/

INSERT INTO sating.historico_coef_rendimento
SELECT
      numero_reg+"-"+dev_reg,
      ano_u_reg,
      mes_u_reg,
      coef_rend_a_reg
FROM fonte.aluno;

UPDATE staging.historico_coef_rendimento
SET periodo = '1'
WHERE periodo<='7';

UPDATE staging.historico_coef_rendimento
SET periodo = '2'
WHERE periodo>'7';
```

I.2) Script de criação e povoamento das entidades do modelo dimensional

```
/*************************************************************

    CRIACAO DA ESTRUTURA DIMENSIONAL
    DO DATA MART  DO SISTEMA DE GRADUACAO

*************************************************************/

/*************************************************************
      DIMENSAO ALUNO
*************************************************************/


CREATE TABLE staging.dim_aluno (
ch_aluno                char(10) PRIMARY KEY NOT NULL,
ch_demografica          number,
ch_espacial             number,
ch_curso                number,
nivel_reg               char(15),
nome_reg                char(40),
ativo_reg               char(10),
convenio_reg            char(3),
insc_vest_reg           char(6),
FAIXA_PONTOS            char(30),
class_vest_reg          char(4),
curso_m_reg             char(20)    DEFAULT NULL,
estado_m_reg            char(2),
ano_conc_m_reg          char(2),
PERIODO_ANO_GRAD        char(13),
ano_trans_reg           char(4),
periodo_trans_reg       char(1),
univers_t_reg           char(20)    DEFAULT NULL,
estado_t_reg            char(2),
FREQUENTA_ALOJAMENTO    char(3)     DEFAULT 'Nao',
NOTA_RCS                char(3)     DEFAULT NULL,
RECEBE_AJUDA            char(3)     DEFAULT 'Nao',
MONITOR                 char(3)     DEFAULT 'Nao'
);


/*************************************************************
      DIMENSAO DISCIPLINA_VESTIBULAR
*************************************************************/


CREATE TABLE staging.dim_disciplina_vestibular (

ch_disciplina_vestibular      char(10),
nome_dist_vest_reg            char(10)
);

CREATE SEQUENCE ch_disciplina_vestibular_seq
INCREMENT BY 1
START WITH 1;
```

```
/*************************************************************
      DIMENSAO CURSO
*************************************************************/
CREATE TABLE staging.dim_curso (

ch_curso    number,
centro      char(15),
unidade     char(15),
curso       char(15),
turno       char(8)
);

CREATE SEQUENCE ch_curso_seq
INCREMENT BY 1
START WITH 1;

/*************************************************************
      DIMENSAO ESPACIAL
*************************************************************/
CREATE TABLE staging.dim_espacial (

ch_espacial       number,
bairro            char(18),
cidade            char(20),
estado            char(2)
);

CREATE SEQUENCE ch_espacial_seq
INCREMENT BY 1
START WITH 1;

/*************************************************************
      DIMENSAO DEMOGRAFICA
*************************************************************/
CREATE TABLE staging.dim_demografica (

ch_demografica    number,
IDADE             number,
sexo              char(1) ,
estado_civil      char(10),
nacionalidade     char(15),
naturalidade      char(15)
);

CREATE SEQUENCE ch_demografica_seq
INCREMENT BY 1
START WITH 1;

/*************************************************************
      DIMENSAO PROFESSOR
*************************************************************/
CREATE TABLE staging.dim_professor (

ch_professor      char(10),
nome_prof         char(30),
sta_prof          char(10)
);
```

```
/*************************************************************
      DIMENSAO TEMPO
*************************************************************/
CREATE TABLE staging.dim_tempo (

ch_tempo    number,
ano         char(4),
periodo     char(1)
);


/*************************************************************
      DIMENSAO VERSAO_DISCIPLINA
*************************************************************/
CREATE TABLE staging.dim_versao_disciplina (

ch_disciplina     char(6),
sta_dis           char(10),
centro_dis        char(15),
unidade_dis       char(15),
departamento_dis  char(15),
nome_dis          char(30),
creditos_dis      char(2)
);

/*************************************************************
      DIMENSAO TURMA
*************************************************************/

CREATE TABLE staging.dim_turma (

numero_turma      char(3),
turma             char(3),
turno             char(8),
centro            char(15),
unidade           char(15),
curso             char(15),
);

/*************************************************************
      TABELA DE FATOS
      CONTROLE_COEF_RENDIMENTO
*************************************************************/
CREATE TABLE staging.dim_controle_coef_rendimento (

ch_curso          number,
ch_aluno          number,
ch_espacial       number,
ch_demografica    number,
ch_tempo          number,
coef_rend_a_reg   char(3)
);
```

```
/*************************************************************
      TABELA DE FATOS
      CONTROLE_VESTIBULAR
*************************************************************/
CREATE TABLE staging.dim_controle_vestibular (

ch_aluno                    number,
ch_curso                    number,
ch_espacial                 number,
ch_demografica              number,
ch_disciplina_vestibular    number,
ch_tempo                    number,
pont_dis_vest_reg           char(5)
);


/*************************************************************
      TABELA DE FATOS
      CONTROLE_DISCIPLINA
*************************************************************/
CREATE TABLE staging.dim_controle_disciplina (

ch_disciplina          number,
ch_tempo               number,
numero_turma           char(3),
ch_professor           number,
numero_alunos_dtr      number,
insc_normal_dtr        number,
insc_tranc_dtr         number,
insc_transf_dtr        number,
insc_irr_dtr           number,
insc_autceg_dtr        number,
insc_32cred_dtr        number,
horas_dadas_dtr        number
);



/*************************************************************
      TABELA DE FATOS
      CONTROLE_NOTAS
*************************************************************/
CREATE TABLE staging.dim_controle_vestibular (

ch_professor     number,
ch_disciplina    number,
ch_tempo         number,
ch_demografica   number,
ch_espacial      number,
ch_aluno         number,
ch_curso         number,
turno            char(8),
conceito_dis_his char(3),
situacao_dis_his char(15)
);
```

```
/*****************************************************************

   POVOAMENTO  DA ESTRUTURA DIMENSIONAL
   DO DATA MART  DO SISTEMA DE GRADUACAO

*****************************************************************/
/*****************************************************************
      DIMENSAO CURSO
*****************************************************************/
INSERT INTO staging.dim_curso
SELECT DISTINCT
      ch_curso_seq.NEXTVAL,
      desc_centro,
      desc_unidade,
      desc_curso,
      'Manhã'
FROM fonte.centro a, fonte.unidade b, fonte.curso c
WHERE c.cod_unidade = b.cod_unidade
AND     b.cod_centro    = a.cod_centro;

INSERT INTO staging.dim_curso
SELECT DISTINCT
      ch_curso_seq.NEXTVAL,
      desc_centro,
      desc_unidade,
      desc_curso,
      'Tarde'
FROM fonte.centro a, fonte.unidade b, fonte.curso c
WHERE c.cod_unidade = b.cod_unidade
AND     b.cod_centro    = a.cod_centro;

INSERT INTO staging.dim_curso
SELECT DISTINCT
      ch_curso_seq.NEXTVAL,
      desc_centro,
      desc_unidade,
      desc_curso,
      'Noite'
FROM fonte.centro a, fonte.unidade b, fonte.curso c
WHERE c.cod_unidade = b.cod_unidade
AND     b.cod_centro    = a.cod_centro;

INSERT INTO staging.dim_curso
SELECT DISTINCT
      ch_curso_seq.NEXTVAL,
      desc_centro,
      desc_unidade,
      desc_curso,
      'Integral'
FROM fonte.centro a, fonte.unidade b, fonte.curso c
WHERE c.cod_unidade = b.cod_unidade
AND     b.cod_centro    = a.cod_centro;
```

```
/*****************************************************************
      DIMENSAO DEMOGRAFICA
*****************************************************************/
INSERT INTO staging.dim_demografica
SELECT DISTINCT
      ch_demografica_seq.NEXTVAL,
      idade,
      sexo,
      estado_civil,
      nacionalidade,
      naturalidade
FROM staging.aluno;

/*****************************************************************
      DIMENSAO ESPACIAL
*****************************************************************/
INSERT INTO staging.dim_espacial
SELECT DISTINCT
      ch_espacial_seq.NEXTVAL,
      bairro,
      cidade,
      estado
FROM staging.aluno;

/*****************************************************************
      DIMENSAO TEMPO
*****************************************************************/
INSERT INTO staging.dim_tempo VALUES (1,'1995','1');
INSERT INTO staging.dim_tempo VALUES (2,'1995','2');
INSERT INTO staging.dim_tempo VALUES (3,'1996','1');
INSERT INTO staging.dim_tempo VALUES (4,'1996','2');
INSERT INTO staging.dim_tempo VALUES (5,'1997','1');
INSERT INTO staging.dim_tempo VALUES (6,'1997','2');
INSERT INTO staging.dim_tempo VALUES (7,'1998','1');
INSERT INTO staging.dim_tempo VALUES (8,'1998','2');
INSERT INTO staging.dim_tempo VALUES (9,'1999','1');
INSERT INTO staging.dim_tempo VALUES (10,'1999','2');
INSERT INTO staging.dim_tempo VALUES (11,'2000','1');
INSERT INTO staging.dim_tempo VALUES (12,'2000','2');
INSERT INTO staging.dim_tempo VALUES (13,'2001','1');
INSERT INTO staging.dim_tempo VALUES (14,'2001','2');
INSERT INTO staging.dim_tempo VALUES (15,'2002','1');
INSERT INTO staging.dim_tempo VALUES (16,'2002','2');
INSERT INTO staging.dim_tempo VALUES (17,'2003','1');
INSERT INTO staging.dim_tempo VALUES (18,'2003','2');
INSERT INTO staging.dim_tempo VALUES (19,'2004','1');
INSERT INTO staging.dim_tempo VALUES (20,'2004','2');
INSERT INTO staging.dim_tempo VALUES (21,'2005','1');
INSERT INTO staging.dim_tempo VALUES (22,'2005','2');
INSERT INTO staging.dim_tempo VALUES (23,'2006','1');
INSERT INTO staging.dim_tempo VALUES (24,'2006','2');
INSERT INTO staging.dim_tempo VALUES (25,'2007','1');
INSERT INTO staging.dim_tempo VALUES (26,'2007','2');
INSERT INTO staging.dim_tempo VALUES (27,'2008','1');
INSERT INTO staging.dim_tempo VALUES (28,'2008','2');
INSERT INTO staging.dim_tempo VALUES (29,'2009','1');
INSERT INTO staging.dim_tempo VALUES (30,'2009','2');
```

```
INSERT INTO staging.dim_tempo VALUES (31,'2010','1');
INSERT INTO staging.dim_tempo VALUES (32,'2010','2');
INSERT INTO staging.dim_tempo VALUES (33,'2011','1');
INSERT INTO staging.dim_tempo VALUES (34,'2011','2');
INSERT INTO staging.dim_tempo VALUES (35,'2012','1');
INSERT INTO staging.dim_tempo VALUES (36,'2012','2');
INSERT INTO staging.dim_tempo VALUES (37,'2013','1');
INSERT INTO staging.dim_tempo VALUES (38,'2013','2');
INSERT INTO staging.dim_tempo VALUES (39,'2014','1');
INSERT INTO staging.dim_tempo VALUES (40,'2014','2');
INSERT INTO staging.dim_tempo VALUES (41,'2015','1');
INSERT INTO staging.dim_tempo VALUES (42,'2015','2');
INSERT INTO staging.dim_tempo VALUES (43,'2016','1');
INSERT INTO staging.dim_tempo VALUES (44,'2016','2');
INSERT INTO staging.dim_tempo VALUES (45,'2017','1');
INSERT INTO staging.dim_tempo VALUES (46,'2017','2');
INSERT INTO staging.dim_tempo VALUES (47,'2018','1');
INSERT INTO staging.dim_tempo VALUES (48,'2018','2');
INSERT INTO staging.dim_tempo VALUES (49,'2019','1');
INSERT INTO staging.dim_tempo VALUES (50,'2019','2');
INSERT INTO staging.dim_tempo VALUES (51,'2020','1');
INSERT INTO staging.dim_tempo VALUES (52,'2020','2');

/*************************************************************
       DIMENSAO DISCIPLINA_VESTIBULAR
*************************************************************/
INSERT INTO staging.dim_disciplina_vestibular
SELECT DISTINCT
      ch_disciplina_vestibular_seq.NEXTVAL,
      nome_disciplina_vestibular
FROM staging.disciplina_vestibular;


/*************************************************************
       DIMENSAO VERSAO_DISCIPLINA
*************************************************************/
INSERT INTO staging.dim_versao_disciplina
SELECT  * FROM staging.versao_disciplina;

/*************************************************************
       DIMENSAO PROFESSOR
*************************************************************/
INSERT INTO staging.dim_professor
SELECT * FROM staging.professor;

/*************************************************************
       DIMENSAO TURMA
*************************************************************/
INSERT INTO staging.dim_turma
SELECT
      numero_turma_dtr,
      turma_dtr,
      turno_dtr,
      centro_dtr,
      unidade_dtr,
      curso_dtr
FROM staging.turma;
```

```
/*************************************************************
       DIMENSAO ALUNO
*************************************************************/
INSERT INTO staging.dim_aluno
SELECT
      a.ch_aluno,
      d.ch_demografica,
      c.ch_espacial,
      b.ch_curso,
      a.nivel_reg,
      a.nome_reg,
      a.ativo_reg,
      a.convenio_reg,
      a.insc_vest_reg,
      a.FAIXA_PONTOS,
      a.class_vest_reg,
      a.curso_m_reg,
      a.estado_m_reg,
      a.ano_conc_m_reg,
      a.PERIODO_ANO_GRAD,
      a.ano_trans_reg,
      a.periodo_trans_reg,
      a.univers_t_reg,
      a.estado_t_reg,
      a.FREQUENTA_ALOJAMENTO,
      a.NOTA_RCS,
      a.RECEBE_AJUDA,
      a.MONITOR,

FROM staging.aluno a, staging.dim_curso b, staging.dim_espacial c,
staging.dim_demografica d
WHERE       a.centro_reg =b.centro
AND   a.unidade_reg=b.unidade
AND   a.curso_reg=b.curso
AND   a.turno_reg=b.turno
AND   a.idade = d.idade
AND   a.sexo_reg = d.sexo
AND   a.estado_civil_reg=d.estado_civil
AND   a.nacionalidade_reg=d.nacionalidade
AND   a.naturalidade_reg=d.naturalidade
AND   a.bairro_reg=c.bairro
AND   a.cidade_reg=c.cidade
AND   a.estado_reg=c.estado


/*************************************************************
       TABELA DE FATOS CONTROLE_COEF_RENDIMENTO
*************************************************************/
INSERT INTO staging.dim_controle_coef_rendimento
SELECT
      a.ch_curso,
      a.ch_aluno,
      a.ch_espacial,
      a.ch_demografica,
      b.ch_tempo
      c.coef_rend_a_reg
```

```
FROM
      staging.dim_aluno a,
      staging.dim_tempo b,
      staging.historico_coef_rendimento c
WHERE
      c.ano = b.ano
AND   c.periodo = b.periodo
AND   a.ch_aluno = c.ch_aluno;

/*************************************************************
       TABELA DE FATOS CONTROLE_VESTIBULAR
*************************************************************/
INSERT INTO staging.dim_controle_vestibular
SELECT
      a.ch_aluno,
      a.ch_curso,
      a.ch_espacial,
      a.ch_demografica,
      b.ch_disciplina_vestibular,
      c.ch_tempo,
      b.pont_dis_vest_reg
FROM
      staging.dim_aluno a,
      staging.disciplina_vestibular b,
      staging.dim_tempo c,
      fonte.aluno d
WHERE
      a.ch_aluno = d.numero_reg+'-'+d.dv_reg
AND   c.periodo = DECODE(d.mes_u_reg,'01','1','02','1','03','1',
                         '04','1','05','1','06','1','07','1','2')
AND   c.ano = d.ano_u_reg
AND   a.ch_aluno = b.ch_aluno;

/* OBS: na comparacao c.ano = d.ano_u_reg, deve ser verificado a questao
de ano com 4 digitos */

/*************************************************************
       TABELA DE FATOS CONTROLE_NOTAS
*************************************************************/
INSERT INTO staging.dim_controle_notas
SELECT
      c.ch_professor,
      a.ch_disciplina,
      d.ch_tempo,
      b.ch_demografica,
      b.ch_espacial,
      b.ch_aluno,
      c.ch_curso,
      c.turno,
      a.conceito_dis_his,
      a.situacao_dis_his
FROM
      staging.registro_notas a,
      staging.dim_aluno b,
      staging.turma c,
      staging.dim_tempo d
WHERE
```

```
      a.ch_aluno = b.ch_aluno
AND   a.ano_his = d.ano
AND   a.periodo_his = d.periodo
AND   a.turma_dtr = c.turma_dtr



/*************************************************************
       TABELA DE FATOS CONTROLE_DISCIPLINA
*************************************************************/
INSERT INTO staging.controle_disciplina
SELECT
      a.ch_disciplina,
      b.ch_tempo,
      a.numero_turma,
      a.ch_professor,
      a.numero_alunos_dtr,
      a.insc_normal_dtr,
      a.insc_tranc_dtr,
      a.insc_transf_dtr,
      a.insc_irr_dtr,
      a.insc_autceg_dtr,
      a.insc_32cred_dtr,
      a.horas_dadas_dtr
FROM
      staging.turma a,
      staging.dim_tempo b,
WHERE
      b.ano = a.ano
AND   b.periodo = a.periodo;
```

II) DM Vestibular

II.1) Script de criação e povoamento das entidades do modelo relacional

```
/*************************************************************

    CRIACAO DA ESTRUTURA RELACIONAL
    DO DATA MART DO SISTEMA DE VESTIBULAR

*************************************************************/

/*************************************************************
      TABELA REGISTRO_VESTIBULANDO
*************************************************************/

CREATE TABLE staging.registro_vestibulando (
matricula_vest                    char(10) PRIMARY KEY NOT NULL,
local_curso_1grau                 char(20),
tipo_escola_1grau                 char(20),
tipo_curso_2grau                  char(20),
tipo_escola_2grau                 char(20),
turno_2grau                       char(10),
mudanca_ultser_2grau              char(20),
```

```
frequencia_cursinho                char(20),
ano_conclusao_2grau                char(20),
pretensao_vestibular               char(20),
exame_vestibulares_anteriores      char(30),
conhecimento_programa_concurso     char(20),
turno_preferencia                  char(20),
posicao_curso_UFRJ                 char(40),
fator_escolha_curso                char(40),
fator_opcao_UFRJ                   char(30),
expectativa_curso_universitario    char(30),
nivel_instrucao_pai                char(30),
nivel_instrucao_mae                char(30),
situacao_trabalho_pai              char(30),
situacao_trabalho_mae              char(30),
ocupacao_pai                       char(30),
ocupacao_mae                       char(30),
renda_mensal_familia               char(30),
participacao_economia_familia      char(30),
pretensao_trabalho_curso           char(20),
numero_pessoas_familia             char(10),
total_dependencias_casa            char(10),
situacao_casa_familia              char(30),
sitio_casapraia_fazenda            char(10),
numero_automoveis                  char(20),
total_livros                       char(30),
leitura_por_ano                    char(30),
lingua_estrangeira                 char(30),
principal_meio_informacao          char(20),
le_jornal                          char(30),
secao_preferida_jornal             char(30),
cursos_extracurriculares           char(30),
acesso_microcomputador             char(20),
utilizacao_microcomputador         char(20),
acesso_internet                    char(20),
faixa_pontos                       char(30),
faixa_classificacao                char(30),
tipo_classificacao                 char(30),
nome                               char(40),
bairro                             char(15),
cidade                             char(15),
estado                             char(02),
idade                              number,
naturalidade                       char(15),
curso_classificado                 char(15),
semestre_classificado              char(01),
turno_classificado                 char(08)
);


/*************************************************************
      TABELA CURSO_OFERECIDO
*************************************************************/

CREATE TABLE staging.curso_oferecido (
cod_curso_oferecido          char(10) PRIMARY KEY NOT NULL,
cod_curso                    char(05),
ano_vestibular               char(04),
```

```
turno                          char(08),
total_vagas                    number,
TOTAL_INSCRITOS_1OPCAO         number,
TOTAL_INSCRITOS_2OPCAO         number,
TOTAL_INSCRITOS_3OPCAO         number,
TOTAL_CLASSIFICADOS            number
);

/*****************************************************************
            TABELA CURSO
*****************************************************************/
CREATE TABLE staging.curso (
cod_curso               char(05) PRIMARY KEY NOT NULL,
nome_curso              char(20),
centro                  char(15),
unidade                 char(15)
);


/*****************************************************************
            TABELA OPCAO_CURSO
*****************************************************************/
CREATE TABLE staging.opcao_curso (
cod_curso_oferecido     char(05),
matricula_vest          char(10),
prioridade_opcao        char(08)
);


/*****************************************************************
            TABELA NOTAS_VESTIBULANDO
*****************************************************************/
CREATE TABLE staging.notas_vestibulando(
cod_disciplina_vestibular     char(08),
matricula_vest                char(10),
nota_disciplina               number
);

/*****************************************************************
            TABELA DISCIPLINA_VESTIBULAR
*****************************************************************/
CREATE TABLE staging.disciplina_vestibular(
cod_disciplina_vestibular     char(08),
descricao_vestibular          char(20)
);
```

```
/*****************************************************************

   POVOAMENTO  DA ESTRUTURA RELACIONAL
   DO DATA MART  DO SISTEMA DE VESTIBULAR

*****************************************************************/


/*****************************************************************
      TABELA REGISTRO_VESTIBULANDO
*****************************************************************/

INSERT INTO staging.registro_vestibulando
(
      matricula_vest,
      local_curso_1grau,
      tipo_escola_1grau
      tipo_curso_2grau
      tipo_escola_2grau
      turno_2grau
      mudanca_ultser_2grau,
      frequencia_cursinho,
      ano_conclusao_2grau,
      pretensao_vestibular,
      exame_vestibulares_anteriores,
      conhecimento_programa_concurso,
      turno_preferencia,
      posicao_curso_UFRJ,
      fator_escolha_curso,
      fator_opcao_UFRJ,
      expectativa_curso_universitario,
      nivel_instrucao_pai,
      nivel_instrucao_mae,
      situacao_trabalho_pai,
      situacao_trabalho_mae,
      ocupacao_pai,
      ocupacao_mae,
      renda_mensal_familia,
      participacao_economia_familia,
      pretensao_trabalho_curso,
      numero_pessoas_familia,
      total_dependencias_casa,
      situacao_casa_familia,
      sitio_casapraia_fazenda,
      numero_automoveis,
      total_livros,
      leitura_por_ano,
      lingua_estrangeira,
      principal_meio_informacao,
      le_jornal,
      secao_preferida_jornal,
      cursos_extracurriculares,
      acesso_microcomputador,
      utilizacao_microcomputador,
      acesso_internet,
      faixa_pontos,
      faixa_classificacao,
```

```
      tipo_classificacao,
      nome,
      bairro,
      cidade,
      estado,
      idade,
      naturalidade
      curso_classificado,
      semestre_classificado,
      turno_classificado
)
SELECT
      a.matricula_vest,
      a.local_curso_1grau,
      a.tipo_escola_1grau
      a.tipo_curso_2grau
      a.tipo_escola_2grau
      a.turno_2grau
      a.mudanca_ultser_2grau,
      a.frequencia_cursinho,
      a.ano_conclusao_2grau,
      a.pretensao_vestibular,
      a.exame_vestibulares_anteriores,
      a.conhecimento_programa_concurso,
      a.turno_preferencia,
      a.posicao_curso_UFRJ,
      a.fator_escolha_curso,
      a.fator_opcao_UFRJ,
      a.expectativa_curso_universitario,
      a.nivel_instrucao_pai,
      a.nivel_instrucao_mae,
      a.situacao_trabalho_pai,
      a.situacao_trabalho_mae,
      a.ocupacao_pai,
      a.ocupacao_mae,
      a.renda_mensal_familia,
      a.participacao_economia_familia,
      a.pretensao_trabalho_curso,
      a.numero_pessoas_familia,
      a.total_dependencias_casa,
      a.situacao_casa_familia,
      a.sitio_casapraia_fazenda,
      a.numero_automoveis,
      a.total_livros,
      a.leitura_por_ano,
      a.lingua_estrangeira,
      a.principal_meio_informacao,
      a.le_jornal,
      a.secao_preferida_jornal,
      a.cursos_extracurriculares,
      a.acesso_microcomputador,
      a.utilizacao_microcomputador,
      a.acesso_internet,
      a.media_vestibular,
      a.classificacao_vestibular,
      a.forma_classificacao,
      b.nome,
```

```
      b.bairro,
      b.cidade,
      b.estado,
      TODAY() - b.data_nascimento,
      b.naturalidade
      a.curso_classificado,
      a.semestre_classificado,
      a.turno_classificado


FROM fonte.questionario a, fonte.vestibulando b
WHERE       a.matricula_vest = b.matricula_vest;


/* CRIANDO CATEGORIAS */

UPDATE staging.registro_vestibulando
SET   faixa_pontos = 'PONTUACAO INFERIOR A 5000'
WHERE       faixa_pontos <= 5000;

UPDATE staging.registro_vestibulando
SET   faixa_pontos = 'PONTUACAO ENTRE 5000 e 7000'
WHERE       faixa_pontos > 5000
AND   faixa_pontos <= 7000;

UPDATE staging.registro_vestibulando
SET   faixa_pontos = 'PONTUACAO ENTRE 7000 e 8000'
WHERE       faixa_pontos > 7000
AND   faixa_pontos <= 8000;

UPDATE staging.registro_vestibulando
SET   faixa_pontos = 'PONTUACAO ACIMA de 8000'
WHERE       faixa_pontos > 8000;


UPDATE staging.registro_vestibulando
SET   faixa_classificacao = 'DEZ PRIMEIROS'
WHERE       faixa_classificacao <= 10;

UPDATE staging.registro_vestibulando
SET   faixa_classificacao = 'ENTRE DECIMO E VIGESIMO'
WHERE       faixa_classificacao > 10
AND   faixa_classificacao <= 20;


UPDATE staging.registro_vestibulando
SET   faixa_classificacao = 'ACIMA DO VIGESIMO'
WHERE       faixa_classificacao > 20;

UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.local_curso_2grau  =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.local_curso_2grau) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='local_curso_2grau';
```

```
UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.tipo_escola_1grau  =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.tipo_escola_1grau) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='tipo_escola_1grau';


UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.tipo_curso_2grau  =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.tipo_curso_2grau) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='tipo_curso_2grau';


UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.tipo_escola_2grau  =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.tipo_escola_2grau) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='tipo_escola_2grau';

UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.turno_2grau  =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.turno_2grau) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='turno_2grau';


UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.mudanca_ultser_2grau  =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.mudanca_ultser_2grau) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='mudanca_ultser_2grau';

UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.frequencia_cursinho  =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.frequencia_cursinho) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='frequencia_cursinho';


UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.ano_conclusao_2grau  =
staging.aux_questionario_de_para.descricao_para
```

```
WHERE TRIM(staging.registro_vestibulando.ano_conclusao_2grau) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='ano_conclusao_2grau';

UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.pretensao_vestibular  =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.pretensao_vestibular) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='pretensao_vestibular';

UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.exame_vestibulares_anteriores =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.exame_vestibulares_anteriores) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo
='exame_vestibulares_anteriores';

UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.conhecimento_programa_concurso =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.conhecimento_programa_concurso)
= staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo
='conhecimento_programa_concurso';

UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.turno_preferencia =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.turno_preferencia) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='turno_preferencia';


UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.posicao_curso_UFRJ =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.posicao_curso_UFRJ) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='posicao_curso_UFRJ';

UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.fator_escolha_curso =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.fator_escolha_curso) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='fator_escolha_curso';
```

```
UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.fator_opcao_UFRJ =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.fator_opcao_UFRJ) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='fator_opcao_UFRJ';


UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.expectativa_curso_universitario =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.expectativa_curso_universitario)
= staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo
='expectativa_curso_universitario';


UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.nivel_instrucao_pai =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.nivel_instrucao_pai) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='nivel_instrucao_pai';

UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.nivel_instrucao_mae =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.nivel_instrucao_mae) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='nivel_instrucao_mae';


UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.situacao_trabalho_pai =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.situacao_trabalho_pai) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='situacao_trabalho_pai';


UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.situacao_trabalho_mae =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.situacao_trabalho_mae) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='situacao_trabalho_mae';
```

```
UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.ocupacao_pai =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.ocupacao_pai) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='ocupacao_pai';


UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.ocupacao_mae =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.ocupacao_mae) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='ocupacao_mae';


UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.renda_mensal_familia =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.renda_mensal_familia) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='renda_mensal_familia';

UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.participacao_economia_familia =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.participacao_economia_familia) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo
='participacao_economia_familia';


UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.pretensao_trabalho_curso =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.pretensao_trabalho_curso) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo
='pretensao_trabalho_curso';

UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.numero_pessoas_familia =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.numero_pessoas_familia) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='numero_pessoas_familia';
```

```
UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.total_dependencias_casa =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.total_dependencias_casa) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='total_dependencias_casa';


UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.situacao_casa_familia =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.situacao_casa_familia) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='situacao_casa_familia';


UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.sitio_casapraia_fazenda =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.sitio_casapraia_fazenda) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='sitio_casapraia_fazenda';


UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.numero_automoveis =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.numero_automoveis) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='numero_automoveis';


UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.total_livros =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.total_livros) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='total_livros';


UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.leitura_por_ano =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.leitura_por_ano) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='leitura_por_ano';
```

```
UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.lingua_estrangeira =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.lingua_estrangeira) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='lingua_estrangeira';

UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.principal_meio_informacao =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.principal_meio_informacao) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo
='principal_meio_informacao';

UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.le_jornal =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.le_jornal) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='le_jornal';

UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.secao_preferida_jornal =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.secao_preferida_jornal) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='secao_preferida_jornal';

UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.cursos_extracurriculares =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.cursos_extracurriculares) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo
='cursos_extracurriculares';

UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.acesso_microcomputador =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.acesso_microcomputador) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='acesso_microcomputador';
```

```
UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.utilizacao_microcomputador =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.utilizacao_microcomputador) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo
='utilizacao_microcomputador';

UPDATE staging.registro_vestibulando
FROM staging.aux_questionario_de_para
SET staging.registro_vestibulando.acesso_internet =
staging.aux_questionario_de_para.descricao_para
WHERE TRIM(staging.registro_vestibulando.acesso_internet) =
staging.aux_questionario_de_para.valor_de
AND staging.aux_questionario_de_para.atributo ='acesso_internet';

/*************************************************************
           TABELA CURSO_OFERECIDO
*************************************************************/

INSERT INTO staging.curso_oferecido (
      cod_curso_oferecido,
      cod_curso,
      ano_vestibular,
      turno,
      total_vagas
)
SELECT
      codigo_curso+'-'+ano_vestibular,
      codigo_curso,
      ano_vestibular,
      turno,
      total_vagas

FROM fonte.vagas_oferecidas;

UPDATE staging.curso_oferecido
SET TOTAL_INSCRITOS_1OPCAO = (
                        SELECT COUNT(fonte.opcao_curso.codigo_curso)
                        FROM fonte.opcao_curso
                        WHERE fonte.opcao_curso.codigo_curso =
                  staging.curso_oferecido.cod_curso_oferecido
                        AND fonte.opcao_curso.prioridade_opcao = 1)
                        );

UPDATE staging.curso_oferecido
SET TOTAL_INSCRITOS_2OPCAO = (
                        SELECT COUNT(fonte.opcao_curso.codigo_curso)
                        FROM fonte.opcao_curso
                        WHERE fonte.opcao_curso.codigo_curso =

staging.curso_oferecido.cod_curso_oferecido
                        AND fonte.opcao_curso.prioridade_opcao = 2)
                        );

UPDATE staging.curso_oferecido
```

```
SET TOTAL_INSCRITOS_3OPCAO = (
                        SELECT COUNT(fonte.opcao_curso.codigo_curso)
                        FROM fonte.opcao_curso
                        WHERE fonte.opcao_curso.codigo_curso =

staging.curso_oferecido.cod_curso_oferecido
                        AND fonte.opcao_curso.prioridade_opcao = 3)
                        );

UPDATE staging.curso_oferecido
SET TOTAL_CLASSIFICADOS = (
                        SELECT
COUNT(staging.registro_vestibulando.matricula_vest)
                        FROM staging.registro_vestibulando
                        WHERE
staging.registro_vestibulando.curso_classificado =
fonte.vagas_oferecidas.cod_curso                                    AND
staging.registro_vestibulando.semestre_classificado=fonte.vagas_oferecida
s.semestre
                    );



/*************************************************************
      TABELA CURSO
*************************************************************/
INSERT INTO staging.curso
SELECT * from fonte.curso;

UPDATE staging.curso
FROM fonte.centro
SET staging.curso.centro = fonte.centro.desc_centro
WHERE staging.curso.centro = fonte.centro.cod_centro;


UPDATE staging.curso
FROM fonte.unidade
SET staging.curso.unidade = fonte.centro.desc_unidade
WHERE staging.curso.unidade = fonte.centro.cod_unidade;

/*************************************************************
            TABELA OPCAO_CURSO
*************************************************************/
INSERT INTO staging.opcao_curso
SELECT      codigo_curso,
      matricula_vest,
      prioridade_opcao
FROM fonte.opcao_curso;

UPDATE staging.opcao_curso
SET prioridade_opcao = 'Primeira'
WHERE prioridade_opcao ='1       ';

UPDATE staging.opcao_curso
SET prioridade_opcao = 'Segunda'
WHERE prioridade_opcao ='2       ';
```

```
UPDATE staging.opcao_curso
SET prioridade_opcao = 'Terceira'
WHERE prioridade_opcao ='3       ';

/*************************************************************
           TABELA NOTAS_VESTIBULANDO
*************************************************************/
INSERT INTO staging.notas_vestibulando
SELECT * FROM fonte.notas_vestibulando;


/*************************************************************
           TABELA DISCIPLINA_VESTIBULAR
*************************************************************/
INSERT INTO staging.disciplina_vestibular
SELECt * FROM fonte.disciplina_vestibular;
```

II.2) Script de criação e povoamento das entidades do modelo dimensional

```
/*************************************************************

    CRIACAO DA ESTRUTURA DIMENSIONAL
    DO DATA MART DO SISTEMA DE VESTIBULAR

*************************************************************/

/*************************************************************
      DIMENSÃO REGISTRO_VESTIBULANDO
*************************************************************/

CREATE TABLE staging.dim_registro_vestibulando (
matricula_vest          char(10),
ch_classificacao        number,
ch_vestibular           number,
ch_socio_economico      number,
ch_ensino               number,
nome                    char(40),
bairro                  char(15),
cidade                  char(15),
estado                  char(02),
idade                   number,
naturalidade            char(15),
curso_classificado      char(15),
semestre_classificado   char(01),
turno_classificado      char(08)
);
```

```
/*************************************************************
      DIMENSÃO CLASSIFICACAO
*************************************************************/

CREATE TABLE staging.dim_classificacao (
ch_classificacao        number,
faixa_pontos            char(30),
faixa_classificacao     char(30),
tipo_classificacao      char(30)
);


CREATE SEQUENCE ch_classificacao_seq
INCREMENT BY 1
START WITH 1;

/*************************************************************
      DIMENSÃO ENSINO
*************************************************************/

CREATE TABLE staging.dim_ensino (
ch_ensino               number,
local_curso_1Grau       char(20),
tipo_escola_1grau       char(20),
tipo_curso_2grau        char(20),
tipo_escola_2grau       char(20),
turno_2grau             char(10),
mudanca_ultser_2grau    char(20),
frequencia_cursinho     char(20),
ano_conclusao_2grau     char(20)
);

CREATE SEQUENCE ch_ensino_seq
INCREMENT BY 1
START WITH 1;

/*************************************************************
      DIMENSÃO VESTIBULAR
*************************************************************/

CREATE TABLE staging.dim_vestibular (
ch_vestibular                       number,
pretensao_vestibular                char(20),
exame_vestibulares_anteriores       char(30),
conhecimento_programa_concurso      char(20),
turno_preferencia                   char(20),
posicao_curso_UFRJ                  char(40),
fator_escolha_curso                 char(40),
fator_opcao_UFRJ                    char(30),
expectativa_curso_universitario     char(30)
);

CREATE SEQUENCE ch_vestibular_seq
INCREMENT BY 1
START WITH 1;
```

```
/*************************************************************
      DIMENSÃO SOCIO_ECONOMICO
*************************************************************/

CREATE TABLE staging.dim_socio_economico (
ch_socio_economico            number,
nivel_instrucao_pai           char(30),
nivel_instrucao_mae           char(30),
situacao_trabalho_pai         char(30),
situacao_trabalho_mae         char(30),
ocupacao_pai                  char(30),
ocupacao_mae                  char(30),
renda_mensal_familia          char(30),
participacao_economia_familia char(30),
pretensao_trabalho_curso      char(20),
numero_pessoas_familia        char(10),
total_dependencias_casa       char(10),
situacao_casa_familia         char(30),
sitio_casapraia_fazenda       char(10),
numero_automoveis             char(20),
total_livros                  char(30),
leitura_por_ano               char(30),
lingua_estrangeira            char(30),
principal_meio_informacao     char(20),
le_jornal                     char(30),
secao_preferida_jornal        char(30),
cursos_extracurriculares      char(30),
acesso_microcomputador        char(20),
utilizacao_microcomputador    char(20),
acesso_internet               char(20),
);

CREATE SEQUENCE ch_socio_economico_seq
INCREMENT BY 1
START WITH 1;

/*************************************************************
      DIMENSAO TEMPO
*************************************************************/
CREATE TABLE staging.dim_tempo (

ch_tempo    number,
ano         char(4)
);


/*************************************************************
      DIMENSAO DISCIPLINA_VESTIBULAR
*************************************************************/


CREATE TABLE staging.dim_disciplina_vestibular (

ch_disciplina_vestibular      char(08),
descricao_disciplina          char(20)
);
```

```
CREATE SEQUENCE ch_disciplina_vestibular_seq
INCREMENT BY 1
START WITH 1;

/*************************************************************
      DIMENSAO CURSO
*************************************************************/
CREATE TABLE staging.dim_curso (
cod_curso         char(05),
nome_curso        char(20),
turno             char(08),
centro            char(15),
unidade           char(15)
);

CREATE SEQUENCE ch_curso_seq
INCREMENT BY 1
START WITH 1;


/*************************************************************
      TABELA DE FATOS CONTROLE_NOTAS_VESTIBULAR
*************************************************************/
CREATE TABLE staging.dim_controle_notas_vestibular (
cod_disciplina_vestibular     char(08),
ch_tempo                      number,
ch_ensino                     number,
ch_vestibular                 number,
ch_socio_economico            number,
ch_classificacao              number,
matricula_vest                char(10),
nota_disciplina               number
);

/*************************************************************
      TABELA DE FATOS CONTROLE_CURSO
*************************************************************/
CREATE TABLE staging.dim_controle_curso (
cod_curso               char(05),
ch_tempo                number,
total_vagas             number,
TOTAL_INSCRITOS_1OPCAO  number,
TOTAL_INSCRITOS_2OPCAO  number,
TOTAL_INSCRITOS_3OPCAO  number,
TOTAL_CLASSIFICADOS     number
);

/*************************************************************
      TABELA DE FATOS CONTROLE_OPCAO
*************************************************************/
CREATE TABLE staging.dim_controle_opcao (
cod_curso               char(05),
matricula_vest          char(10),
ch_tempo                number,
prioridade_opcao        char(08)
);
```

```
/*************************************************************

   POVOAMENTO DA ESTRUTURA RELACIONAL
   DO DATA MART DO SISTEMA DE VESTIBULAR

*************************************************************/

/*************************************************************
      DIMENSÃO CLASSIFICACAO
*************************************************************/
INSERT INTO staging.dim_classificacao
SELECT DISTINCT
      ch_classificacao_seq.NEXTVAL,
      faixa_pontos,
      faixa_classificacao,
      tipo_classificacao
FROM staging.registro_vestibulando;


/*************************************************************
      DIMENSÃO ENSINO
*************************************************************/
INSERT INTO staging.dim_ensino
SELECT DISTINCT
      ch_ensino_seq.NEXTVAL,
      local_curso_1Grau,
      tipo_escola_1grau,
      tipo_curso_2grau,
      tipo_escola_2grau,
      turno_2grau,
      mudanca_ultser_2grau,
      frequencia_cursinho,
      ano_conclusao_2grau
FROM staging.registro_vestibulando;

/*************************************************************
      DIMENSÃO VESTIBULAR
*************************************************************/

INSERT INTO staging.dim_vestibular
SELECT DISTINCT
      ch_vestibular_seq.NEXTVAL,
      pretensao_vestibular,
      exame_vestibulares_anteriores,
      conhecimento_programa_concurso,
      turno_preferencia,
      posicao_curso_UFRJ,
      fator_escolha_curso,
      fator_opcao_UFRJ,
      expectativa_curso_universitario
FROM staging.registro_vestibulando;


/*************************************************************
      DIMENSÃO SOCIO_ECONOMICO
*************************************************************/
INSERT INTO staging.dim_socio_economico
```

```
SELECT
      ch_socio_economico_seq.NEXTVAL,
      nivel_instrucao_pai,
      nivel_instrucao_mae,
      situacao_trabalho_pai,
      situacao_trabalho_mae,
      ocupacao_pai,
      ocupacao_mae,
      renda_mensal_familia,
      participacao_economia_familia,
      pretensao_trabalho_curso,
      numero_pessoas_familia,
      total_dependencias_casa,
      situacao_casa_familia,
      sitio_casapraia_fazenda,
      numero_automoveis,
      total_livros,
      leitura_por_ano,
      lingua_estrangeira,
      principal_meio_informacao,
      le_jornal,
      secao_preferida_jornal,
      cursos_extracurriculares,
      acesso_microcomputador,
      utilizacao_microcomputador,
      acesso_internet
FROM staging.registro_vestibulando;

/*************************************************************
      DIMENSÃO REGISTRO_VESTIBULANDO
*************************************************************/
INSERT INTO staging.dim_registro_vestibulando
SELECT
      a.matricula_vest,
      e.ch_classificacao,
      b.ch_vestibular,
      c.ch_socio_economico,
      d.ch_ensino,
      a.nome,
      a.bairro,
      a.cidade,
      a.estado,
      a.idade,
      a.naturalidade,
      a.curso_classificado,
      a.semestre_classificado,
      a.turno_classificado
FROM
      staging.registro_vestibulando a,
      staging.dim_vestibular b,
      staging.dim_socio_economico c,
      staging.dim_ensino d,
      staging.dim_classificacao e
WHERE
      a.local_curso_1grau = d.local_curso_1grau
AND   a.tipo_escola_1grau = d.tipo_escola_1grau
AND   a.tipo_curso_2grau = d.tipo_curso_2grau
```

```
AND   a.tipo_escola_2grau = d.tipo_escola_2grau
AND   a.turno_2grau      = d.turno_2grau
AND   a.mudanca_ultser_2grau = d.mudanca_ultser_2grau
AND   a.frequencia_cursinho = d.frequencia_cursinho
AND   a.ano_conclusao_2grau = d.ano_conclusao_2grau
AND   a.pretensao_vestibular = b.pretensao_vestibular
AND   a.exame_vestibulares_anteriores  = b.exame_vestibulares_anteriores
AND   a.conhecimento_programa_concurso  =
b.conhecimento_programa_concurso
AND   a.turno_preferencia = b.turno_preferencia
AND   a.posicao_curso_UFRJ = b.posicao_curso_UFRJ
AND   a.fator_escolha_curso = b.fator_escolha_curso
AND   a.fator_opcao_UFRJ = b.fator_opcao_UFRJ
AND   a.expectativa_curso_universitario =
b.expectativa_curso_universitario
AND   a.nivel_instrucao_pai = c.nivel_instrucao_pai
AND   a.nivel_instrucao_mae = c.nivel_instrucao_mae
AND   a.situacao_trabalho_pai = c.situacao_trabalho_pai
AND   a.situacao_trabalho_mae = c.situacao_trabalho_mae
AND   a.ocupacao_pai = c.ocupacao_pai
AND   a.ocupacao_mae = c.ocupacao_mae
AND   a.renda_mensal_familia = c.renda_mensal_familia
AND   a.participacao_economia_familia = c.participacao_economia_familia
AND   a.pretensao_trabalho_curso = c.pretensao_trabalho_curso
AND   a.numero_pessoas_familia = c.numero_pessoas_familia
AND   a.total_dependencias_casa = c.total_dependencias_casa
AND   a.situacao_casa_familia = c.situacao_casa_familia
AND   a.sitio_casapraia_fazenda = c.sitio_casapraia_fazenda
AND   a.numero_automoveis = c.numero_automoveis
AND   a.total_livros = c.total_livros
AND   a.leitura_por_ano = c.leitura_por_ano
AND   a.lingua_estrangeira = c.lingua_estrangeira
AND   a.principal_meio_informacao = c.principal_meio_informacao
AND   a.le_jornal = c.le_jornal
AND   a.secao_preferida_jornal = c.secao_preferida_jornal
AND   a.cursos_extracurriculares = c.cursos_extracurriculares
AND   a.acesso_microcomputador = c.acesso_microcomputador
AND   a.utilizacao_microcomputador = c.utilizacao_microcomputador
AND   a.acesso_internet = c.acesso_internet
AND   a.faixa_pontos = e.faixa_pontos
AND   a.faixa_classificacao = e.faixa_classificacao
AND   a.tipo_classificacao = e.tipo_classificacao


/*************************************************************
      DIMENSAO TEMPO
*************************************************************/
insert into staging.tempo values (1,1990);
insert into staging.tempo values (2,1991);
insert into staging.tempo values (3,1992);
insert into staging.tempo values (4,1993);
insert into staging.tempo values (5,1994);
insert into staging.tempo values (6,1995);
insert into staging.tempo values (7,1996);
insert into staging.tempo values (8,1997);
insert into staging.tempo values (9,1998);
insert into staging.tempo values (10,1999);
```

```
insert into staging.tempo values (11,2000);
insert into staging.tempo values (12,2001);
insert into staging.tempo values (13,2002);
insert into staging.tempo values (14,2003);
insert into staging.tempo values (15,2004);
insert into staging.tempo values (16,2005);
insert into staging.tempo values (17,2006);
insert into staging.tempo values (18,2007);
insert into staging.tempo values (19,2008);
insert into staging.tempo values (20,2009);
insert into staging.tempo values (21,2010);
insert into staging.tempo values (22,2011);
insert into staging.tempo values (23,2012);
insert into staging.tempo values (24,2013);
insert into staging.tempo values (25,2014);
insert into staging.tempo values (26,2015);
insert into staging.tempo values (27,2016);
insert into staging.tempo values (28,2017);
insert into staging.tempo values (29,2018);
insert into staging.tempo values (30,2019);
insert into staging.tempo values (31,2020);

/*************************************************************
      DIMENSAO DISCIPLINA_VESTIBULAR
*************************************************************/

INSERT INTO staging.dim_disciplina_vestibular
SELECT
      ch_disciplina_vestibular,
      descricao_disciplina
FROM staging.disciplina_vestibular

/*************************************************************
      DIMENSAO CURSO
*************************************************************/
INSERT INTO staging.dim_curso
SELECT DISTINCT
      ch_curso_seq.NEXTVAL,
      desc_centro,
      desc_unidade,
      desc_curso,
      'Manhã'
FROM fonte.centro a, fonte.unidade b, fonte.curso c
WHERE c.cod_unidade = b.cod_unidade
AND     b.cod_centro    = a.cod_centro;

INSERT INTO staging.dim_curso
SELECT DISTINCT
      ch_curso_seq.NEXTVAL,
      desc_centro,
      desc_unidade,
      desc_curso,
      'Tarde'
FROM fonte.centro a, fonte.unidade b, fonte.curso c
WHERE c.cod_unidade = b.cod_unidade
AND     b.cod_centro    = a.cod_centro;
```

```
INSERT INTO staging.dim_curso
SELECT DISTINCT
      ch_curso_seq.NEXTVAL,
      desc_centro,
      desc_unidade,
      desc_curso,
      'Noite'
FROM fonte.centro a, fonte.unidade b, fonte.curso c
WHERE c.cod_unidade = b.cod_unidade
AND     b.cod_centro    = a.cod_centro;

INSERT INTO staging.dim_curso
SELECT DISTINCT
      ch_curso_seq.NEXTVAL,
      desc_centro,
      desc_unidade,
      desc_curso,
      'Integral'
FROM fonte.centro a, fonte.unidade b, fonte.curso c
WHERE c.cod_unidade = b.cod_unidade
AND     b.cod_centro    = a.cod_centro;


/*************************************************************
      TABELA DE FATOS CONTROLE_NOTAS_VESTIBULAR
*************************************************************/
INSERT INTO staging.dim_controle_notas_vestibular
SELECT
      a.cod_disciplina_vestibular,
      c.ch_tempo,
      b.ch_ensino,
      b.ch_vestibular   ,
      b.ch_socio_economico,
      b.ch_classificacao,
      a.matricula_vest,
      a.nota_disciplina
FROM
      staging.notas_vestibulando a,
      staging.dim_registro_vestibulando b,
      staging.dim_tempo c,
      fonte.opcao_curso d
WHERE
      a.matricula_vest = b.matricula_vest
AND   a.matricula_vest = d.matricula_vest
AND   d.prioridade_opcao = '1'
AND   d.ano_vestibular = c.ano;

/*************************************************************
      TABELA DE FATOS CONTROLE_CURSO
*************************************************************/
INSERT INTO staging.dim_controle_curso
SELECT
      a.cod_curso,
      b.ch_tempo,
      a.total_vagas,
      a.TOTAL_INSCRITOS_1OPCAO,
      a.TOTAL_INSCRITOS_2OPCAO,
```

```
      a.TOTAL_INSCRITOS_3OPCAO,
      a.TOTAL_CLASSIFICADOS
FROM
      staging.curso_oferecido a,
      staging.dim_tempo b
WHERE
      a.ano_vestibular = b.ano


/*************************************************************
      TABELA DE FATOS CONTROLE_OPCAO
*************************************************************/
INSERT INTO staging.dim_controle_opcao
SELECT
      a.cod_curso_oferecido,
      a.matricula_vest,
      b.ch_tempo,
      a.prioridade_opcao
FROM
      staging.opcao_curso a,
      staging.dim_tempo b
      staging.curso_oferecido c
WHERE
      a.cod_curso_oferecido = c.cod_curso_oferecido
AND   a.cod_curso = c.cod_curso
AND   c.ano_vestibular = b.ano
```

Anexo 3

CATEGORIAS, ARTEFATOS, DEFINIÇÕES DE PADRÕES, VALORES "DEFAULT" E REGRAS DE CONVERSÃO

3.1 – Data Mart Graduação

a) Categoria IDADE

Definido como a subtração da data atual pela data de nascimento do registro do Aluno na tabela do banco de dados fonte.

b) Categoria FAIXA_PONTOS_VEST

Construída de acordo com o valor de pontos_vest_reg da entidade Aluno do banco de dados fonte:

Se pontos_vest_reg < 5000,
então faixa_pontos_vest = 'PONTUACAO INFERIOR A 5000'

Se pontos_vest_reg >= 5000 e pontos_vest_reg < 7000,
então faixa_pontos_vest = 'PONTUACAO ENTRE 5000 e 7000'

Se pontos_vest_reg >= 7000 e pontos_vest_reg < 8000,
então faixa_pontos_vest = 'PONTUACAO ENTRE 7000 e 8000'

Se pontos_vest_reg >= 8000,
então faixa_pontos_vest = 'PONTUACAO ACIMA DE 8000'

c) Categoria PERIODO_ANO_GRAD

Construída de acordo com o valor do campo mes_g_reg e ano_g_reg do registro do Aluno, na tabela do banco de dados fonte:

Se mes_g_reg > 7, então PERIODO_ANO_GRAD = '2PERIODO '+ano_g_reg

Senão, PERIODO_ANO_GRAD = ‘1PERIODO’+ano_g_reg

d) Artefato FREQUENTA_ALOJAMENTO

Criado, com valor default “Não”, os registros de Alunos que existirem na tabela ALOJAMENTO do banco de dados fonte, terão seu valor atualizado para “Sim”.

e) Artefato RECEBE_AJUDA

Criado, com valor default “Não”, os registros de Alunos que existirem na tabela AJUDA_CUSTO do banco de dados fonte, terão seu valor atualizado para “Sim”.

f) Artefato MONITOR

Criado, com valor default “Não”, os registros de Alunos que existirem na tabela MONITOR do banco de dados fonte, terão seu valor atualizado para “Sim”.

g) Artefato NOTA_RCS

Criado buscando-se o valor do campo NOTA_RCS da tabela ALUNO do banco de dados fonte.

h) Definição de padrões, Valores “Default” e regras de conversão

Entidade *Aluno*	
Atributo	**Regras para Conversão**
Nivel_reg	**1,2,3 → graduação,** **4 → extensão,** **5 → aperfeiçoamento,** **6 → especialização,** **7 → mestrado,** **8 →doutorado,** **9→ pós-doutorado**
centro_reg	**Substituir código pela descrição**
unidade_reg	**Substituir código pela descrição**
curso_reg	**Substituir código pela descrição**
sexo_reg	**"0" → "M"** **"1" → "F"**
Nacionalidade_reg	**1 → brasileiro,** **2 → Naturalizado,** **3 → Estrangeiro**
Naturalidade_reg	**Substituir código pela descrição**
ativo_reg	**"A" → ativa,** **"T" → trancada,** **"C"→ cancelada**
curso_m_reg	**Substituir código pela descrição**
univers_t_reg	**Substituir código pela descrição**

Nota_RCS	-----

Entidade *Versão_Disciplina*	
Atributo	**Regras para a Conversão**
centro_dis	**Substituir código pela descrição**
unidade_dis	**Substituir código pela descrição**
curso_dis	**Substituir código pela descrição**

Entidade *Turma*	
Atributo	**Regras para a Conversão**
centro_dtr	**Substituir código pela descrição**
unidade_dtr	**Substituir código pela descrição**
curso_dtr	**Substituir código pela descrição**

Entidade *Professor*	
Atributo	**Regras para a Conversão**
sta_prof	**"A" → ativa,** **"T" → trancada,** **"C"→ cancelada**

3.2 – Data Mart Vestibular

a) Categoria IDADE

Definido como a subtração da data atual pela data de nascimento do registro do Vestibulando na tabela do banco de dados fonte.

b) Categoria FAIXA_PONTOS

Construída de acordo com o valor de pontos_vestibular da entidade Registro_Vestibulando do banco de dados fonte:

Se pontos_vestibular < 5000,
então faixa_pontos = ‘PONTUACAO INFERIOR A 5000’

Se pontos_vestibular >= 5000 e pontos_vestibular < 7000,
então faixa_pontos = ‘PONTUACAO ENTRE 5000 e 7000’

Se pontos_vestibular >= 7000 e pontos_vestibular < 8000,
então faixa_pontos = ‘PONTUACAO ENTRE 7000 e 8000’

Se pontos_vestibular >= 8000,
então faixa_pontos = ‘PONTUACAO ACIMA DE 8000’

c) Categoria FAIXA_CLASSIFICACAO

Construída de acordo com o valor do campo classificacao_vestibular do registro do vestibulando, na tabela REGISTRO_VESTIBULANDO do banco de dados fonte:

Se classificacao_vestibular < 10,
então faixa_classificacao = ‘DEZ PRIMEIROS’

Se classificacao_vestibular > 10 e classificacao_vestibular <= 20,
então faixa_classificacao = ‘ENTRE DECIMO E VIGESIMO’

Se classificacao_vestibular > 20,
então faixa_classificacao = ‘ACIMA DO VIGESIMO’

d) Definição de padrões, Valores “Default” e regras de conversão

Entidade *Vestibulando*	
Atributo	**Regras para a Conversão**
Curso_classificado	**Substituir código pela descrição**
tipo_classificacao	0 → NÃO CLASSIFICADO 1 → 1ª CLASSIFICAÇÃO 2 → 1ª RECLASSIFICAÇÃO 2 → 2ª RECLASSIFICAÇÃO

Entidade *Questionário*	
Atributo	**Regras para a Conversão**
local_curso_2Grau	A → Cidade do Rio de Janeiro B → Outra Cidade do RJ C → Região SE – RJ D → Região S E → Região N e CO F → Região NE G → Exterior
tipo_escola_1grau	A → Escola Pública B → Escola Particular C → Maior Parte Pública

	D → Maior Parte Particular
tipo_curso_2grau	A → Atual 2° Grau B → Técnico C → Magistério – Antigo Normal D → Supletivo E → Outro
tipo_escola_2grau	A → Escola Pública B → Escola Particular C → Maior Parte Pública D → Maior Parte Particular
turno_2grau	A → Manhã B → Tarde C → Noite D → Integral
mudanca_ultser_2grau	A → Não B → Sim. Escola Conceituada C → Sim. Localização D → Sim. Financeira E → Sim. Outras
frequencia_cursinho	A → Não B → Sim. Semestre C → Sim. Ano D → Sim. Mais de um Ano
ano_conclusao_2grau	A → Ano atual B → Ano anterior C → Dois anos antes D → Três anos antes E → Quatro ou mais
pretensao_vestibular	A → Único B → Outros com mesma opção C → Outros com outras opções
exame_vestibulares_anterio-res	A → Não B → Sim. Sem classificação curso desejado C → Sim. Com classificação / sem instituição D → Sim. Mudou idéia do curso E → Sim. Problemas financeiros F → Sim. Outros
conhecimento_programa_ concurso	A → Desconhece programas B → Ouviu falar C → Conhece, mas não emprega D → Estuda por eles
turno_preferencia	A → Noite B → Diurno, aceita Noite C → Diurno, não aceita Noite
posicao_curso_UFRJ	A → Curso inscrito B → Outros cursos da mesma área C → Qualquer curso da mesma área D → Determinado curso outra área E → Qualquer Curso / qualquer área
fator_escolha_curso	A → Mercado de Trabalho B → Prestígio social da profissão C → Adequação aptidões pessoais D → Baixa concorrência vagas E → Amplas possibilidades salariais
fator_opcao_UFRJ	A → Única com o curso

	B → Melhor curso C → Melhor horário D → Pouco procurada – fácil classificação E → Mais fácil acesso F → Seguir opção amigos
expectativa_curso_ universitario	A → Cultura geral ampla B → Voltado ao Mercado de Trabalho C → Voltado para pesquisa D → Formação acadêmica – ativ. prática E → Compreender melhor o mundo F → Melhorar nível de instrução
nivel_instrucao_pai	A → Nenhum B → Menos que 4ª série 1º Grau C → 4ª série 1º Grau D → Mais 4ª série 1º Grau e Menos que 8ª série E → 1º Grau completo F → 2º Grau incompleto G → 2º Grau completo H → Superior incompleto I → Superior completo
nivel_instrucao_mae	A → Nenhum B → Menos que 4ª série 1º Grau C → 4ª série 1º Grau D → Mais 4ª série 1º Grau e Menos que 8ª série E → 1º Grau completo F → 2º Grau incompleto G → 2º Grau completo H → Superior incompleto I → Superior completo
situacao_trabalho_pai	A → Empregado B → Desempregado C → Aposentado D → Vive de renda E → Falecido F → Não tem informação
situacao_trabalho_mae	A → Empregada B → Desempregada C → Aposentada D → Vive de renda E → Falecida F → Não tem informação
ocupacao_pai	A → Empresário B → Proprietário C → Executivo D → Ocupação de nível superior E → Ocupação de nível médio F → Ocupação manual G → Trabalhador rural
ocupacao_mae	A → Empresária B → Proprietária C → Executiva D → Ocupação de nível superior E → Ocupação de nível médio F → Ocupação manual

	G → Trabalhadora rural
renda_mensal_familia	A → Até 1 SM B → Maior que 1 SM a 3 SM C → Maior que 3 SM a 5 SM D → Maior que 5 SM a 10 SM E → Maior que 10 SM a 20 SM F → Maior que 20 SM a 30 SM G → Maior que 30 SM
participacao_economia_ familia	A → Gastos financiados família B → Trabalha e auxílio financeiro C → Trabalha e auto-sustenta D → Trabalha e contribui p/ família E → Trabalha e responsável p/ família
pretensao_trabalho_curso	A→ Não B → Sim. Estágio C → Sim. Últimos anos D → Sim. Em tempo parcial E → Sim. Em tempo integral
numero_pessoas_familia	A → Vive só B → Duas C → Três D → Quatro E → Cinco F → Seis ou mais
total_dependencias_casa	A → 1 ou 2 B → 3 C → 4 D → 5 E → Mais de cinco
situacao_casa_familia	A → Própria e quitada B → Própria e não quitada C → Alugada D → Outra forma de ocupação
sitio_casapraia_fazenda	A → Sim B → Não
numero_automoveis	A → Não tem B → Tem 1 C → Tem 2 D → Tem mais de 2
total_livros	A → Nenhum B → Até 20 C → de 21 a 50 D → de 51 a 100 E → de 101 a 200 F → de 201 a 500 G → mais de 500
leitura_por_ano	A → Nenhuma B → 1 a 2 C → 3 a 5 D → 6 a 10 F → 11 ou mais
lingua_estrangeira	A → Sim. Fluentemente B → Sim. Razoavelmente C → Não. Gostaria de aprender D → Não. Sem necessidade
principal_meio_informacao	A → Jornal

	B → Televisão C → Internet D → Rádio E → Revista F → Outras pessoas G → Não se atualiza
le_jornal	A → Não B → Sim. Ocasionalmente C → Sim. Todos os domingos D → Sim. Diariamente
secao_preferida_jornal	A → Política B → Economia C → Esporte D → Cultura E → Notícias internacionais F → Notícias locais G → Quadrinho H → Ciência I → Informática J → Outros assuntos
cursos_extracurriculares	A → Não B → Sim. Línguas estrangeiras C → Sim. Ginástica/balé/esportes D → Sim. Música/arte E → Sim. Outros
acesso_microcomputador	A → Não B → Sim. Em casa C → Sim. Outros locais
utilizacao_microcomputador	A → Não usa B → Trabalhos escolares C → Jogos D → Fins profissionais E → Outros
acesso_internet	A → Não B → Sim. Em casa C → Sim. Outros locais D → Desconhece Internet

Entidade *Curso*	
Atributo	**Regras para a Conversão**
Centro	**Substituir código pela descrição**
Unidade	**Substituir código pela descrição**

REFERÊNCIAS BIBLIOGRÁFICAS

BERSON, Alex, SMITH, Smith J. **Data Warehousing, Data Mining, & OLAP.** New York. McGraw-Hill, 1997. 612 p.

CAMPOS, Maria Luiza, FILHO, A.V. Rocha. **Data Warehouse.** In: XVII Congresso da Sociedade Brasileira de Computação – XVI Jornada de Atualização em Informática. Brasília, 1997, pg 221-261.

FIRESTONE, Joseph M. **Architectural Evolution in Data Warehousing and Distribuited Knowledge Management Architecture**. Disponível na INTERNET via http://www.dkms.com/ARCHEV.html. Arquivo consultado em 2000.

GOODYEAR, Mark, RYAN, Hugh W., SARGENT, Scott R., et al. **Netcentric and Client/Server Computing – a practical guide. Andersen Consulting.** Boca Raton: Auerbach, 1999.

GUPTA, Vivek R. **An Introdution to Data Warehousing.** Disponível na INTERNET via http://www.system-services.com/dwintro.html . Arquivo consultado em 1999.

HUFFORD, Duane. **Data Warehouse Quality**. Disponível na INTERNET via http://www.datawarehouse.com/master.cfm. Arquivo consultado em 1999.

INMON, W.H. .**Como construir o Data Warehouse.** Rio de Janeiro: Editora Campus, 1997. 388p.

INMOM, W.H, HACKATHORN, Richard D. **Como Usar o Data Warehouse.** Rio de Janeiro: Editora Infobook, 1997. 277p.

INMON, W.H., IMHOFF, Claudia, BATTAS, Greg. **Building the Operation Data Store.** New York: John Wiley & Sons, Inc., 1998

KIMBALL, Ralph. **Data Warehouse Toolkit.** São Paulo: Makron Books, 1998. 388 p.

_____. **The Aggregate Navigator.** Disponível na INTERNET via http://www.dbmsmag.com/9511d05.html. Arquivo consultado em 1998. (a)

_____. **Aggregate Navigation with (almost) no Metadata.** Disponível na INTERNET via http://www.dbmsmag.com/9608d54.html. Arquivo consultado em 1998. (b)

_____. **Mastering Data Extraction.** Disponível na INTERNET via http://www.dbmsmag.com/9606d05.html. Arquivo consultado em 1999.

KIMBALL, Ralph, REEVES, Laura, ROSS, Margy, THORNTHWAITE, Warren. **The Data Warehouse Lifecycle Toolkit – Expert Methods for Designing, Developing and Deploying Data Warehouses**. New York: John Wiley & Sons, Inc., 1998. 771 p.

McGUFF, Frank. **Designing the Perfect Data Warehouse.** Disponível na INTERNET via http://members.aol.com/fmcguff/dwmodel/frtext.htm. Arquivo consultado em 2000.

MELO, Rubens Nascimento. **Data Warehousing (Tutorial).** In: XIII SBBD - Simpósio Brasileiro de Banco de Dados, Salvador, 1997.

MEYER, Don, CANNON, Casey. **Building a Better Data Warehouse.** New Jersey: Prentice-Hall PTR, 1998. 227 p.

PLATINUM. **Creating the No Compromise Warehouse.** Disponível na INTERNET via http://www.platinum.com/products/wp/wp_dbase.htm. Arquivo consultado em 1999.

POE, Vidette, KLAUER, Patricia, BROBST, Stephen. **Building a Data Warehouse for Decision Support.** New Jersey. Prentice-Hall, Inc, 1998. 285 p.

SOARES, Vânia Jesus de Araujo. **Modelagem Incremental no Ambiente de Data Warehouse.** Rio de Janeiro: UFRJ/IM/NCE, 1998. 216p. Dissertação (Mestrado)