WBA0749_v1.0

INTEGRAÇÃO E FLUXO DE DADOS (ETL)

Dados Internacionais de Catalogação na Publicação (CIP)

Alves, Thiago Salhab
A474i Integração e fluxo de dados (ETL)/ Thiago Salhab Alves,–
Londrina: Editora e Distribuidora Educacional S.A. 2019.
140 p.
ISBN 978-85-522-1497-7
1. Banco de dados. 2. big data.
I. Alves, Thiago Salhab. II. Título.

CDD 000

Responsável pela ficha catalográfica: Thamiris Mantovani CRB-8/9491

2019
Editora e Distribuidora Educacional S.A.
Avenida Paris, 675 – Parque Residencial João Piza
CEP: 86041-100 — Londrina — PR
e-mail: editora.educacional@kroton.com.br
Homepage: http://www.kroton.com.br/

SUMÁRIO

Apresentação da disciplina

A disciplina de Integração e Fluxo de Dados inicia com a compreensão das definições e dos conceitos básicos dos *data warehouses* e processos de extração, transformação e carga (ETL – Extract Transform Load).

Na aula 2, são trabalhadas as ferramentas de ETL e o Microsoft SQL Server Integration Services (SSIS), sua arquitetura e seus componentes.

Na aula 3, são trabalhados as definições e os conceitos básicos de extração de dados, aprendendo a criar um mapa de dados lógico e conhecer os diferentes tipos de dados de origem que podem ser extraídos.

Na aula 4, são trabalhados as definições e os conceitos de limpeza de dados, aprendendo a remover ruídos e dados irrelevantes e a corrigir inconsistências nos dados, aprendendo as técnicas de compartimentalização (*binning*), regressão e agrupamento.

Na aula 5, são trabalhados as definições e os conceitos de conformação de dados, aprendendo a tratar problemas como redundância e conflito de valores e técnicas de consolidação de dados.

Na aula 6, são trabalhados o processo de entrega de dados, dimensões planas e dimensões flocos de neve, dimensões de data e hora e dimensões grandes.

Para finalizar, na aula 7, são trabalhados os conceitos de ETL de Tempo Real, aplicações e abordagens de ETL de Tempo Real.

Introdução à extração, transformação e carga (ETL)

Autor: Thiago Salhab Alves

Objetivos

- Compreender as definições e os conceitos básicos dos *data warehouses*.
- Aprender sobre os processos de extração, transformação e carga (ETL – Extract Transform Load).
- Aprender sobre arquiteturas de *data warehousing*.

1. *Data Warehouse* e ETL

Prezado aluno, você já parou para pensar como as organizações, nacionais e multinacionais, tomariam suas decisões diárias sem o uso de sistemas de apoio à tomada de decisões? Como seria o dia a dia de um diretor ou presidente de uma multinacional sem o uso de um sistema de DDS (Decision Support System) ou BI (Business Intelligence)? Certamente, o processo de tomada de decisão seria muito complicado. Dessa forma, faz-se necessário o uso de um *data warehouse* para contribuir com processos decisórios mais importantes da organização.

Esta leitura irá apresentar as definições e os conceitos básicos dos *data warehouses*. Você aprenderá sobre os processos de extração, transformação e carga (ETL), bem como arquiteturas de *data warehouses*. Uma boa aula e bom trabalho.

1.1 *Data Warehouse*

De acordo com Kimball e Caserta (2009), por mais de 30 anos se utilizaram aplicações de banco de dados, porém é difícil obter esses dados para fins analíticos. Investimentos de bilhões de dólares já foram gastos em aplicativos de banco de dados, mas os dados armazenados nesses bancos são, em sua grande maioria, de difícil manipulação para fins analíticos e para uso na tomada de decisões.

Segundo Turban et al. (2009), um *data warehouse* é um conjunto de dados utilizado no suporte à tomada de decisões, sendo um repositório de dados atuais e históricos, orientado por assunto, integrado, variável no tempo e não volátil. De acordo com Kimball e Caserta (2009), um *data warehouse* tem por objetivo publicar dados da organização com suporte mais efetivo para a tomada de decisão, tendo como critério principal para o sucesso contribuir para os processos decisórios mais importantes da organização.

Os custos de um *data warehouse*, gerenciado pelo TI, são táticos, mas os custos e benefícios mais importantes do suporte à decisão são estratégicos. Os dados estão estruturados de forma a estarem disponíveis em um formato pronto para ser utilizado em atividades de processamento analítico, como, por exemplo, processamento analítico on-line (OLAP), *data mining*, consultas, geração de relatórios, etc.

Turban et al. (2009) apresentam a definição de que *data warehouse* é o repositório de dados e *data warehousing* é o processo inteiro, sendo uma disciplina que resulta em aplicações que oferecem suporte à tomada de decisão, permitindo acesso imediato às informações de negócios. Kimball e Caserta (2009) definem que *data warehouse* é o processo de obter dados de sistemas de bancos de dados legados e de transações e transformá-los em informações organizadas em um formato amigável para incentivar a análise de dados e apoiar a tomada de decisões beseadas em fatos.

Um *data warehouse* é um sistema que extrai, limpa, conforma e entrega fonte de dados em armazenamento de dados dimensional e implementa consultas e análises para fins de tomada de decisão (KIMBALL; CASERTA, 2009).

Segundo Turban et al. (2009), são características fundamentais dos *data warehouses*:

- Orientados por assunto: os dados devem ser organizados por assunto detalhado, tais como vendas, produtos ou clientes, contendo apenas informações relevantes ao suporte à decisão. A orientação por assunto de um *data warehouse* proporciona uma visão mais abrangente da organização.
- Integrados: os dados de diferentes fontes devem estar em um formato consistente e totalmente integrados nos *data warehouses*.

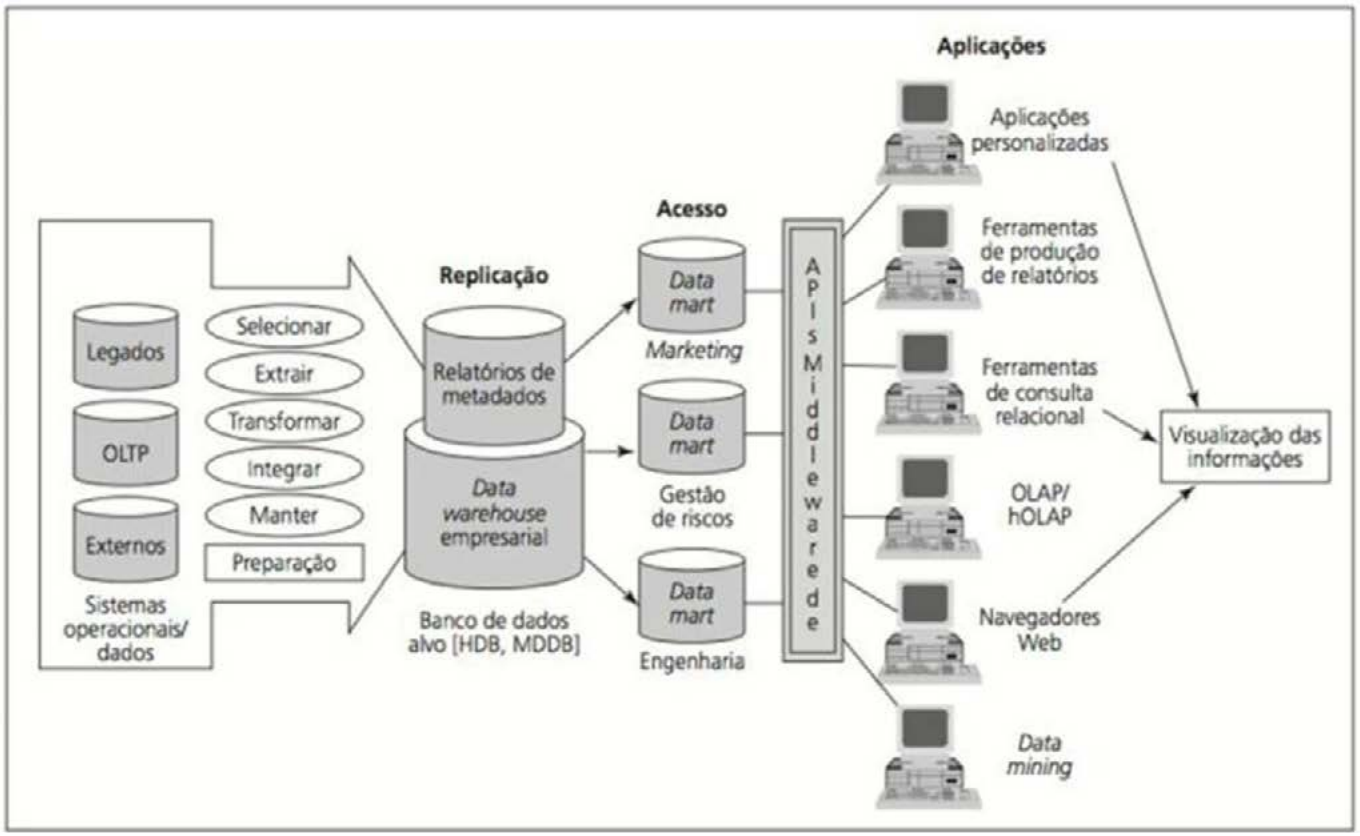
- Variáveis no tempo: o *data warehouse* deve manter dados históricos. Os dados permitem detectar tendências, variações, relações de longo prazo para previsão e comparações, levando à tomada de decisões. Existe uma qualidade temporal para cada *data warehouse*, sendo o tempo uma importante dimensão, podendo conter diversos pontos de tempo, como, por exemplo, visualizações diárias, semanais e mensais.
- Não voláteis: após os dados serem inseridos no *data warehouse*, os usuários não podem alterá-los.

De acordo com Turban et al. (2009), há três tipos de *data warehouses*:

- *Data mart:* é um subconjunto de um *data warehouse* que consiste em uma única área temática, como, por exemplo, marketing e operações, sendo um *warehouse* pequeno, projetado para uma unidade estratégica de negócios ou departamento.
- *Data store operacional:* permite arquivar informações recentes para consumo, sendo um banco de dados usado constantemente na área de preparação temporária de um *data warehouse*. Os conteúdos do *data store operacional*, ao contrário dos conteúdos estáticos dos *data warehouses*, são atualizados durante o curso das operações comerciais.
- *Data warehouses empresariais:* é um *data warehouse* em grande escala, usado por toda a empresa no suporte à decisão. Oferece interação dos dados de muitas fontes, em um formato padronizado.

A Figura 1 ilustra o conceito de *data warehouse*, em que os dados são importados de vários recursos internos (legados e OLTP) e externos, passando os dados pelos processos de extração, transformação, integração e preparação, para, em seguida, preencher o *data warehouse* com dados.

Figura 1 – Conceito de *data warehouse*



Fonte: TURBAN et al., 2009, p. 61.

Data marts podem ser carregados para uma área ou um departamento específicos (marketing, gestão de riscos e engenharia). As ferramentas de *middleware* permitem acesso ao *data warehouse* e os usuários podem escrever suas próprias consultas SQL e utilizar em muitas aplicações *front-end*, incluindo *data mining*, OLAP, ferramentas de relatórios e visualização de dados.

Como exemplo dos benefícios da utilização de um *data warehouse*, Turban et al. (2009) relatam que a empresa aérea Continental Airlines apresentou em 1994 sérios problemas financeiros, tendo suas vendas de passagens prejudicadas por atrasos nas partidas dos voos, problemas na chegada de bagagens e *overbooking*. A Continental, em 1998, possuía bancos de dados separados para marketing e operações, hospedados e gerenciados por fornecedores externos, com o processamento de consultas e os programas de marketing para clientes demorado e ineficiente.

Como processo de solução, foi realizado um processo de integração das fontes de dados operacionais, de marketing, de TI e de receita em um único *data warehouse*, oferecendo diversos benefícios para a Continental, que voltou a ter lucros.

ASSIMILE

Segundo Kimball e Caserta (2009), o principal critério para o sucesso de um *data warehouse* é se ele irá contribuir efetivamente para os processos decisórios da empresa. Embora custos tangíveis do projeto de um *data warehouse* (hardware, software, mão de obra) devam ser geridos com cuidado, os custos intangíveis de não se apoiarem decisões importantes na organização são potencialmente muito maiores que os tangíveis.

1.2 Extração, transformação e carga (ETL)

De acordo com Turban et al. (2009), a parte fundamental do processo de *data warehouse* é a extração, transformação e carga (ETL). O processo de ETL é componente integral de qualquer projeto que utiliza dados e é um grande desafio para os gerentes de TI, pois o processo para se obter, limpar, consolidar e transformar os dados consome 70% do tempo do projeto.

O processo de ETL consiste na extração (leitura dos dados de um ou mais bancos de dados), transformação (conversão dos dados extraídos de sua forma original na forma que precisam estar para serem inseridos em um *data warehouse*) e carga (colocação dos dados no *data warehouse*) (TURBAN et al., 2009).

De acordo com Devmedia (2019), o processo ETL (Extract, Transform and Load) é um processo que exige esforço e a maior parte do tempo de construção de um *data warehouse*. Esse processo vai extrair dados de fontes de dados heterogêneas e tem que alimentar o *data warehouse* de

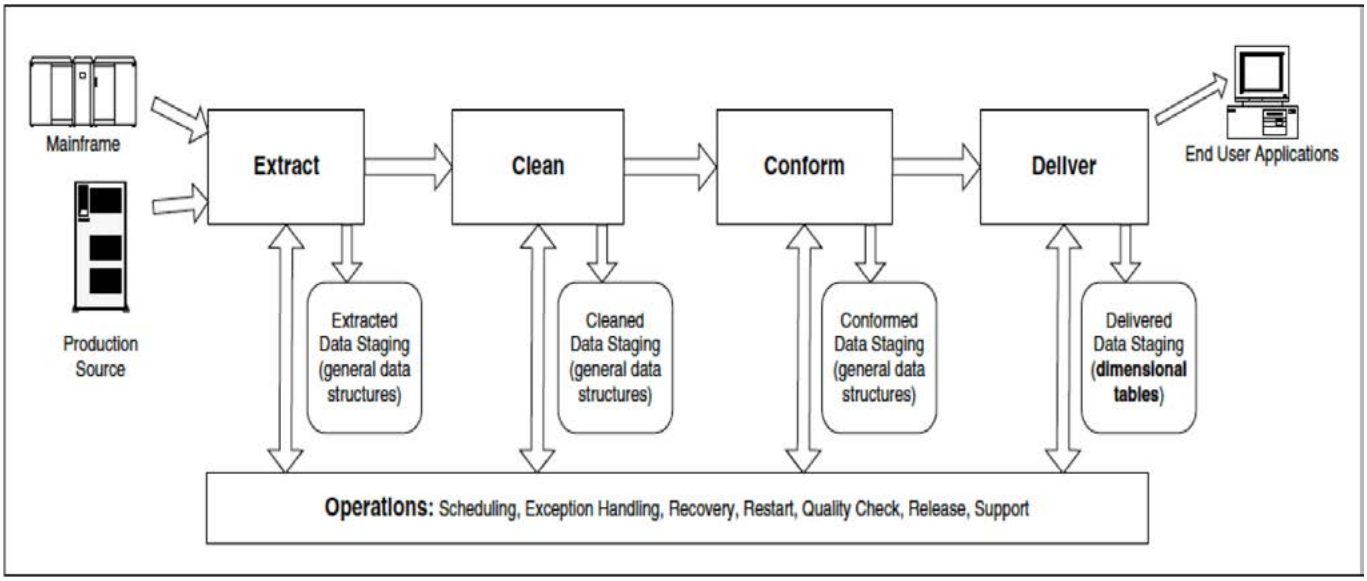
forma homogênea e concisa, pois vai servir de base para gerar relatórios e gráficos de apoio à decisão para a gerência da corporação e não pode trazer resultados errôneos.

Segundo o Devmedia (2009), após selecionar os dados que serão carregados no *data warehouse*, vem a parte de tratamento ou transformação e limpeza dos dados, que consiste em padronizar os dados com relação ao tamanho e tipo, substituição de caracteres estranhos, correção de erros de digitação, etc. O processo de carga é a parte em que a ferramenta de ETL vai carregar os dados no *data warehouse* para que sejam utilizados.

Segundo Turban et al. (2009), a ETL é extremamente importante na integração de dados e tem por objetivo carregar dados integrados e limpos no *data warehouse*, dados que podem vir de qualquer fonte, como, por exemplo, aplicação de *mainframe*, um ERP (Enterprise Resource Planning), ferramenta de CRM (Customer Relationship Management), arquivo de texto ou planilha eletrônica.

A Figura 2 ilustra o processo de extração, transformação e carga (ETL) de dados. A fonte de dados apresenta dados que foram extraídos, limpos, transformados e carregados para serem utilizados como apoio à tomada de decisões.

Figura 2 – Processo de ETL

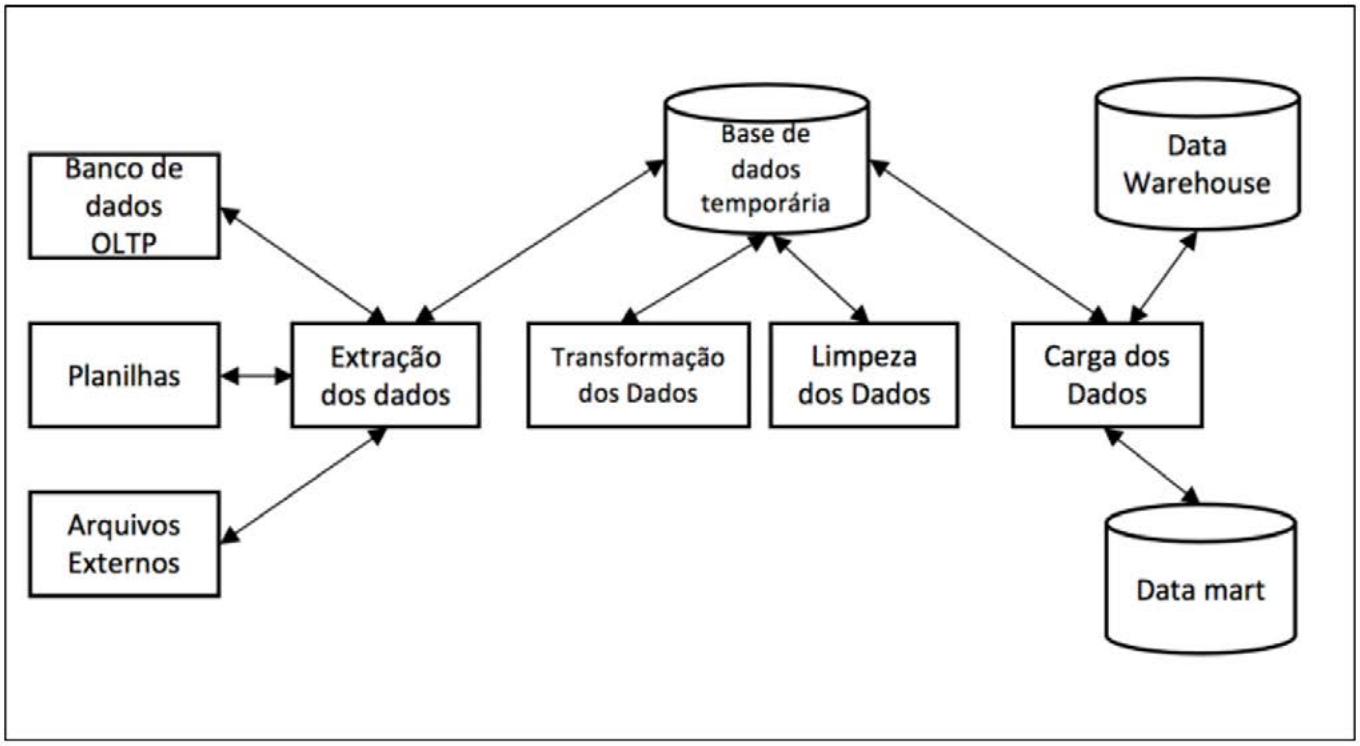


Fonte: KIMBALL; CASERTA, 2009, p. 18.

O processo de migração de dados para o *data warehouse*, de acordo com Turban et al. (2009), consiste na extração de dados de várias fontes relevantes, a limpeza, transformação e o carregamento dos dados, cujo processo está descrito na Figura 3. As fontes são arquivos extraídos de bancos de dados OLTP (*Online Transaction Processing*), planilhas, banco de dados ou arquivos externos. Os arquivos de entrada são gravados em um conjunto de tabelas temporárias, criadas para facilitar o processo de carga.

A Figura 3 ilustra os elementos utilizados no processo de extração, transformação e carga dos dados para o *data warehouse*. A fonte de dados pode ser um banco de dados, planilhas ou arquivos externos. Uma base de dados temporária é utilizada enquanto os dados são extraídos, transformados, limpos e carregados para o *data warehouse ou data mart*.

Figura 3 – Elementos de ETL



Fonte: o autor.

De acordo com Kimball e Caserta (2009), na extração, os dados não tratados vindos de sistemas de origem são gravados em disco e dados estruturados são gravados em arquivos simples ou tabelas relacionais.

Na limpeza, o nível de qualidade aceitável para sistemas de origem é diferente da qualidade de dados necessária para o *data warehouse*. Alguns passos devem ser seguidos, tais como checar valores válidos, consistência dos valores, remover valores duplicados.

Na conformação, os dados são conformados quando duas ou mais origens de dados são mescladas no *data warehouse*. Fontes de dados separadas não podem ser consultadas juntas, a menos que alguns ou todos os rótulos dessas fontes sejam idênticas ou similares. Na entrega, o objetivo é deixar os dados prontos para consulta. Consiste em estruturar os dados em um conjunto de esquemas simples e simétricos, conhecidos como modelos dimensionais.

Segundo Turban et al. (2009), o processo de carregar dados para um *data warehouse* pode ser realizado por meio de ferramentas de transformação de dados, que oferecem uma GUI (Graphical User Interface) para ajudar no desenvolvimento e na manutenção das regras de negócio. O processo de carregamento de dados também pode ser executado de forma tradicional, com o desenvolvimento de programas ou utilitários, usando linguagens de programação, para que seja realizado o carregamento do *data warehouse*.

Várias são as questões que vão determinar a aquisição de uma ferramenta de transformação de dados ou a construção de sua própria ferramenta:

- Ferramentas de transformação de dados são caras;
- Ferramentas de transformação de dados têm uma curva de aprendizado longa;
- Ferramentas de transformação de dados necessitam de um aprendizado sólido de uso para poderem medir o desempenho de uma organização.

Como exemplo de uma ETL eficiente, Turban et al. (2009) relatam que a Motorola Inc. usa a ETL para carregar dados em seus *data warehouses*, coletando informações de mais de 30 diferentes sistemas de aquisição, e os envia para o seu *data warehouse* global de SCM (Supply Chain Management) para uma análise dos gastos agregados da empresa. Alguns critérios, de acordo com Turban et al. (2009), devem ser utilizados para selecionar a ferramenta de ETL:

- Capacidade de ler e gravar um número ilimitado de arquiteturas de fontes de dados;
- Captura e entrega automática de metadados;
- Histórico de conformidade com padrões abertos;
- Interface fácil de usar para o desenvolvedor e usuário final.

De acordo com Turban et al. (2009), antes do uso dos *data warehouses*, *data marts* e dos softwares de BI, permitir o acesso a fontes de dados era trabalhoso, pois mesmo com as ferramentas de gestão de dados, saber quais dados acessar e oferecê-los ao tomador de decisão era uma tarefa complicada e exigia especialistas em bancos de dados.

PARA SABER MAIS

De acordo com Kimball e Caserta (2009), a missão de uma equipe ETL é possibilitar:

- Entregar dados de maneira mais eficaz às ferramentas do usuário final;
- Agregar valor aos dados nas etapas de limpeza e conformidade;
- Proteger e documentar os dados.

De acordo com Kimball e Caserta (2009), alguns tipos de estruturas de dados que são necessários para o sistema ETL:

- Arquivos simples: ao utilizar uma ferramenta ETL ou utilizar *shell scripts* ou programas de script, tais como Perl, VBScript ou JavaScript, que podem manipular arquivos de texto, é possível armazenar dados de preparação em seu sistema de arquivos como arquivos de texto simples. Se os dados estiverem armazenados em linhas e colunas em arquivos que emulem uma tabela de banco de dados, ela é utilizada como arquivo simples ou sequencial. Os arquivos simples podem ser processados e manipulados por ferramentas de ETL ou linguagens de script, como se fossem tabelas de banco de dados e, em certos casos, de forma mais rápida que tabelas de banco de dados.
- Conjunto de dados XML: XML é uma linguagem para comunicação de dados e pode ser usado na forma de documentos de texto simples contendo dados e metadados. O XML permite declarar estruturas hierárquicas e, quando o *data warehouse* recebe dados XML definidos, utiliza-se um processo de extração para transferir os dados para um *data warehouse* relacional.
- Tabelas relacionais: os dados de preparação podem ser armazenados usando tabelas de banco de dados, sendo mais apropriado quando se tem uma ferramenta ETL dedicada, apresentando algumas vantagens, tais como suporte DBA e interface SQL.
- Tabelas independentes: usar tabelas que não apresentam dependências de qualquer outra tabela da base de dados.
- Fontes de dados não relacionais: alguns projetos de *data warehouse* possuem dados de fontes de dados não relacionais, podendo ser arquivos simples, planilhas, etc.
- Modelos de dados dimensionais: estruturas de dados dimensionais são utilizadas nos processos de ETL, sendo as estruturas mais populares para consulta, simples de criar, estáveis e rápidas para consulta de banco de dados relacional.

- Tabelas de fatos: uma medida é uma observação do mundo real de um valor desconhecido, sendo as medições predominantemente numéricas e podendo ser repetidas ao longo do tempo. Uma única medida cria um único registro da tabela de fatos que correspondem a um evento de medição específico.

1.3 Arquiteturas de *data warehouse*

Segundo Turban et al. (2009), há algumas arquiteturas básicas de *data warehousing*, sendo as arquiteturas de duas e três camadas as mais comuns. As arquiteturas são divididas em três partes:

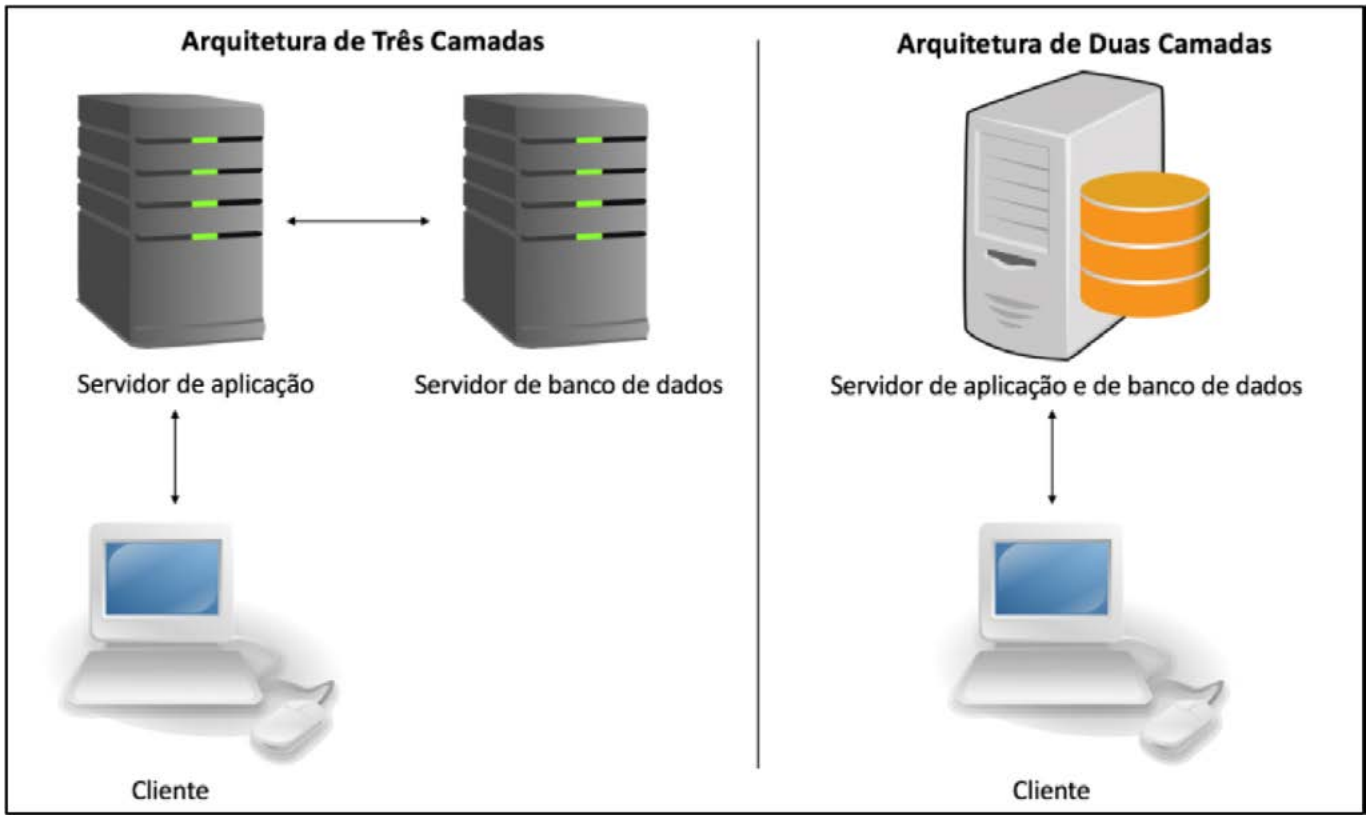
- *Data warehouse*, contendo os dados e softwares associados;
- Software de aquisição de dados, que extrai dados dos sistemas de entrada, consolida-os e depois os carrega no *data warehouse*;
- Software cliente (*front-end*), que permite que os usuários acessem e analisem os dados a partir do *warehouse*, utilizando um mecanismo de DDS (Decision Support System) ou BI (Business Intelligence).

Na arquitetura de três camadas, os sistemas operacionais contêm os dados e o software para aquisição em uma camada (servidor), o *data warehouse* em outra camada e a terceira camada inclui o mecanismo de DSS/ BI (servidor de aplicação) e o cliente. A vantagem de usar a arquitetura de três camadas é poder separar funções do *data warehouse*, eliminando as limitações de recursos e possibilitar a criação de *data marts*.

Na arquitetura de duas camadas, de acordo com Turban et al. (2009), o mecanismo de DSS/ BI ou BA é executado fisicamente na mesma plataforma de hardware que o *data warehouse*, sendo mais econômico que a arquitetura de três camadas, mas pode apresentar problemas de desempenho em caso de *data warehouses* grandes que manipulem aplicações de alto desempenho no suporte à tomada de decisão.

A Figura 4 ilustra as arquiteturas de duas e três camadas. Na arquitetura de três camadas, há o cliente, o servidor de aplicação (que pode incluir mecanismos de DSS/BI ou BA) e o *data warehouse* no servidor de banco de dados. Na arquitetura de duas camadas, o servidor de aplicação e o *data warehouse* ficam na mesma plataforma de hardware.

Figura 4 – Arquitetura *data warehouse* com duas e três camadas

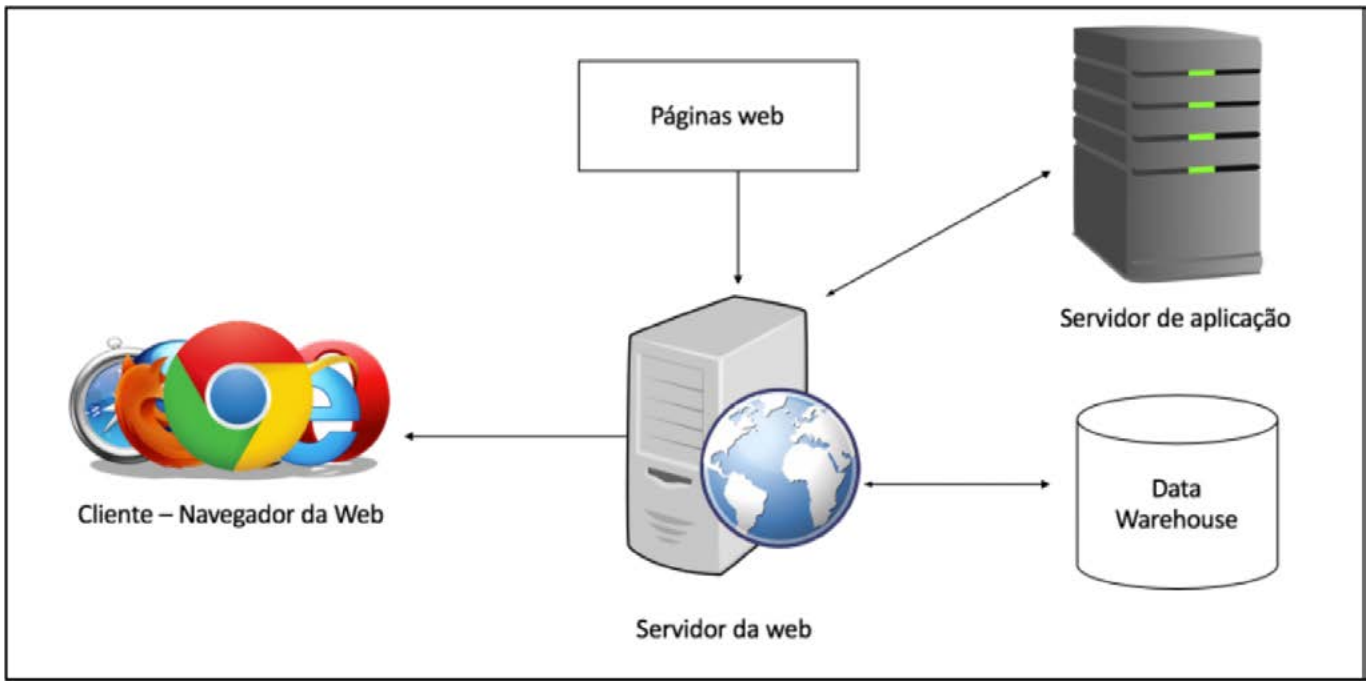


Fonte: o autor.

O *data warehouse* integrado à internet, segundo Turban et al. (2009), produz o *data warehousing* baseado na web. Ela utiliza três camadas e inclui cliente, servidor web e servidor de aplicação. Do lado cliente, o usuário necessita de conexão à internet e navegador web, e do lado servidor, um servidor web para administrar os fluxos de entrada e saída de informação, e o *data warehouse*. O *data warehousing* baseado na web oferece como vantagens a facilidade de acesso, independência de plataforma e baixo custo.

A Figura 5 ilustra a arquitetura do *data warehouse* baseado na web, uma arquitetura que possui três camadas e inclui o cliente com seu navegador web, servidor da web e servidor da aplicação.

Figura 5 – Arquitetura *data warehouse* baseado na web



Fonte: o autor.

Para a escolha de que arquitetura utilizar, algumas questões devem ser analisadas:

- Qual sistema de gerenciamento de banco de dados (SGBD) usar?

  A maioria dos *data warehouses* usam SGBD relacional, tais como Oracle, SQL Server e DB2, que são produtos que suportam arquiteturas cliente/servidor e baseado na web.

- O processamento será paralelo e/ou os dados serão particionados?

  Processamento paralelo permite que múltiplas CPUs processem solicitações de consultas ao *data warehouse* simultaneamente e oferece escalabilidade. Os projetistas também devem decidir se as tabelas de bancos de dados serão particionadas (divididas em tabelas menores) para uma maior eficiência de acesso.

Dessa forma, você aprendeu as definições e os conceitos básicos dos *data warehouses*, bem como os processos de extração, transformação e carga (ETL) e arquiteturas de *data warehousing*.

TEORIA EM PRÁTICA

Uma empresa nacional de revenda de cosméticos está enfrentando alguns problemas financeiros. Dado o grande volume de produtos lançados pelo setor de cosméticos, a empresa está tendo dificuldades em acompanhar a demanda por produtos lançados, o que, muitas vezes, acaba por comprometer o resultado financeiro, por investir em produtos com baixa procura. Outro problema são os produtos que possuem prazo de validade curto, que acabam por vencer e não podem ser trocados pelos fornecedores. Hoje, a empresa conta com um sistema de vendas e controle de estoque, com banco de dados relacional e um processo de marketing pelas redes sociais, porém está tendo dificuldades para a tomada de decisões relacionadas ao que o seu público-alvo realmente consome e assim evitar gastos desnecessários. Como podemos auxiliar a organização a reverter essa situação?

VERIFICAÇÃO DE LEITURA

1. Um *data warehouse* é um conjunto de dados utilizado no suporte à tomada de decisões, sendo um repositório de dados atuais e históricos, orientado por assunto, integrado, variável no tempo e não volátil. Assinale a alternativa que apresente, corretamente, as partes fundamentais do processo de um *data warehouse*.

   a. Extração, transformação e carga.

   b. Extração, separação e carga.

c. Extração, mineração e transformação.

d. Extração, separação e transformação.

e. Extração, mineração e carga.

2. O processo de migração de dados para o *data warehouse*, de acordo com Turban et al. (2009), consiste na extração de dados de várias fontes relevantes, limpeza, transformação e carregamento dos dados. Os arquivos de entrada são gravados em um conjunto de tabelas temporárias. Assinale a alternativa que apresenta, corretamente, a função dessas tabelas temporárias.

a. Facilitar o processo de extração.

b. Facilitar o processo de transformação.

c. Facilitar o processo de limpeza.

d. Facilitar o processo de carga.

e. Facilitar o processo de mineração.

3. O *data warehousing* utiliza algumas arquiteturas básicas. Assinale a alternativa que apresente, corretamente, essas arquiteturas.

a. Arquitetura de uma camada.

b. Arquitetura de uma e duas camadas.

c. Arquitetura de duas e três camadas.

d. Arquitetura de uma e três camadas.

e. Arquitetura de duas e quatro camadas.

Referências bibliográficas

DEVMEDIA. Extract, **Transformation and Load (ETL)** – Ferramentas BI. Disponível em: https://www.devmedia.com.br/extract-transformation-and-load-etl-ferramentas-bi/24408. Acesso em: 28 abr. 2019.

KIMBALL, R.; CASERTA, J. **The *Data Warehouse* ETL Toolkit:** practical techniques for extracting, cleaning, conforming, and data delivering data. Indianopolis: Wiley Publishing, 2009.

TURBAN, E. et al. **Business Intelligence:** um enfoque gerencial para a inteligência do negócio. Porto Alegre: Bookman, 2009.

Gabarito

Questão 1 – Resposta A

As partes fundamentais do processo de um *data warehouse* são extração, transformação e carga.

Questão 2 – Resposta D

A função das tabelas temporárias é facilitar o processo de carga.

Questão 3 – Resposta C

O processo de *data warehouse* utiliza algumas arquiteturas básicas que são a arquitetura de duas e três camadas.

Ferramentas de ETL

Autor: Thiago Salhab Alves

Objetivos

- Conhecer as ferramentas de ETL.
- Conhcer a ferramenta Microsoft SQL Server Integration Services (SSIS).
- Conhecer a arquitetura e os componentes da ferramenta Microsoft SQL Server Integration Services (SSIS).

1. Ferramentas de ETL

Prezado aluno, você já parou para pensar na quantidade de dados que as organizações nacionais e multinacionais manipulam diariamente? Muitos sistemas necessitam de dados “limpos” para poder usar em seus sistemas de apoio à tomada de decisões e as ferramentas de ETL (Extract, Transform and Load) fornecem um apoio fundamental na preparação desses dados para uso. Dessa forma, faz-se necessário conhecer as ferramentas de ETL e suas propostas.

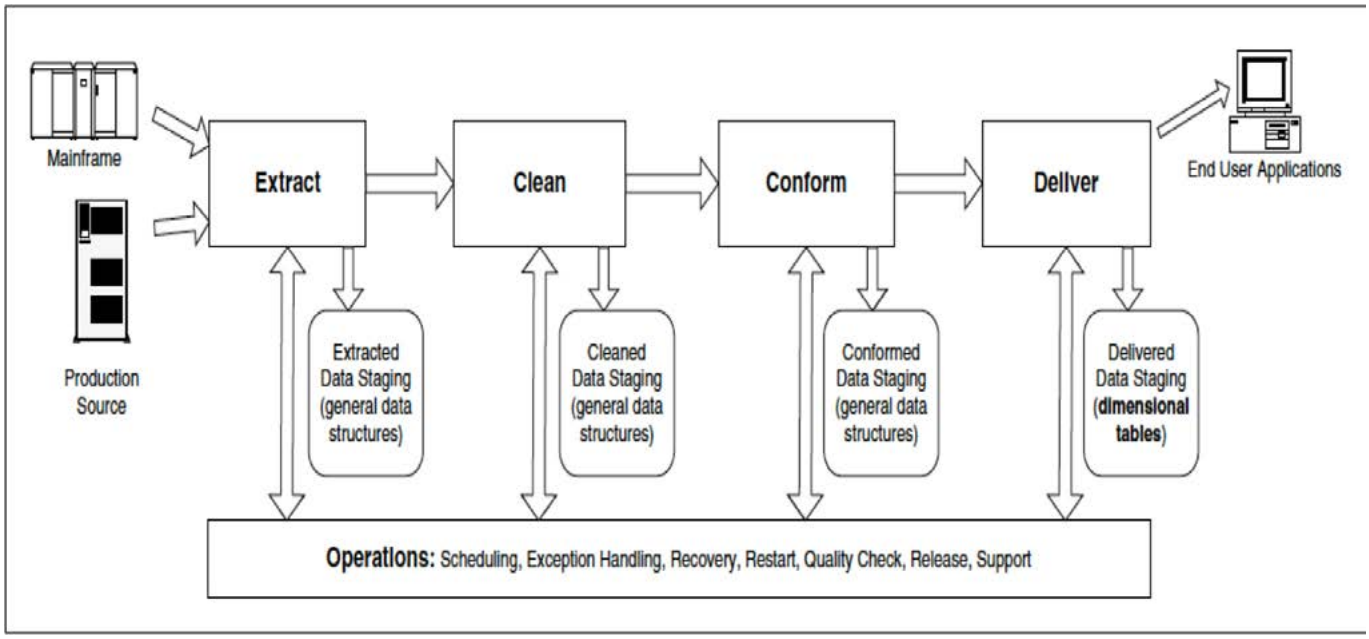
Esta leitura irá apresentar algumas ferramentas de ETL. Você conhecerá a ferramenta Microsoft SQL Server Integration Services (SSIS), bem como sua arquitetura e seus componentes. Uma boa aula e bom trabalho.

1.1 Ferramentas de ETL

De acordo com Pall e Khaira (2013), o processo de integração de dados envolve práticas, técnicas arquiteturais e ferramentas para obter acesso consistente e para entregar dados em uma grande área de assuntos e tipos de estrutura em uma empresa. A demanda do mercado se torna cada vez mais diversificada e os usuários buscam ferramentas que apoiem iniciativa de Business Intelligence.

O processo de carregar dados para um *data warehouse* pode ser realizado por meio de ferramentas de transformação de dados, que oferecem uma GUI (Graphical User Interface) para ajudar no desenvolvimento e na manutenção das regras de negócio. A Figura 6 ilustra o processo de ETL (Extract, Transform and Load), atividade essencial que extrai dados de diversas fontes de dados, tais como base de dados, web sites, documentos e ferramentas de software, que transforma esses dados em dados limpos e carrega o *data warehouse*, que entrega dados para web sites, *e-commerces*, ferramentas de software, entre outros sistemas.

Figura 6 – Processo de ETL



Fonte: KIMBALL; CASERTA, 2009, p. 18.

Várias são as questões que vão determinar a aquisição de uma ferramenta de transformação de dados ou a construção de sua própria ferramenta:

- Ferramentas de transformação de dados são caras;
- Ferramentas de transformação de dados têm uma curva de aprendizado longa;
- Medir o desempenho da organização de TI não é tarefa trivial de ser realizada até que a empresa aprenda a usar as ferramentas de transformação de dados.

PARA SABER MAIS

O investimento em integração de dados tem aumentado a cada dia e começa a fazer parte do orçamento que a organização reserva a cada ano. A demanda do mercado torna-se mais diversificada à medida que os compradores adquirem ferramentas com a intenção de dar suporte a vários casos de uso.

Alguns critérios, de acordo com Turban et al. (2009), devem ser utilizados para selecionar a ferramenta de ETL:

- Capacidade de ler e gravar um número ilimitado de arquiteturas de fontes de dados;
- Captura e entrega automática de metadados;
- Histórico de conformidade com padrões abertos;
- Interface fácil de usar para o desenvolvedor e usuário final.

De acordo com Pall e Khaira (2013), as ferramentas de ETL são categorizadas em:

- ETL codificado manualmente;
- ETL baseado em ferramenta.

As ferramentas de ETL codificadas manualmente são desenvolvidas em algumas linguagens de programação, tais como Perl, Cobol, C e PL/SQL, para extrair dados de múltiplas fontes de arquivos, transformar esses dados e carregá-los na base de dados. Esses programas são longos e difíceis de documentar. O desenvolvedor necessita usar diferentes linguagens de programação para executar as tarefas de ETL, tais como scripts em Perl para extrair dados das fontes do sistema e executar transformações e SQL para carregar os dados no *data warehouse*.

As ferramentas ETL codificadas manualmente possuem a vantagem de criar metadados que podem ser gerenciados diretamente e dar flexibilidade ao desenvolver para manipular suas necessidades, e como desvantagens, para que se atenda a essas mudanças nos grandes volumes de dados gerados de várias fontes, os programas precisam ser modificados com frequência, causando um impacto no projeto como um todo e o ETL codificado manualmente geralmente é lento na execução, uma vez que é de encadeamento único.

O ETL baseado em ferramenta, segundo Pall e Khaira (2013), iniciou suas extrações em *mainframes* para bancos de dados de destino. Essas são as ferramentas ETL que hoje fornecem características de transformação, suportam vários arquivos de banco de dados de entrada ou saída, projetos multidimensionais, várias funções de transformação e banco de dados nativo, eliminando a sobrecarga de desenvolvimento e manutenção de rotinas e transformações complexas em fluxos de trabalho de ETL.

Essas ferramentas estão fornecendo GUIs amigáveis ao usuário, o que permite que o desenvolvedor trabalhe com recursos como monitoramento, programação, carregamento em massa, agregação incremental, etc. Segundo Pall e Khaira (2013), as ferramentas ETL podem ser classificadas em quatro subcategorias:

- Ferramentas ETL puras: são produtos independentes da base de dados e da ferramenta de Business Intelligence que se pretende usar. As empresas não precisam depender de nenhum outro produto para a funcionalidade oferecida e permitem a migração para diferentes bases de dados sem mudar o processo de integração.
- Ferramenta ETL de base de dados integrada: são produtos fornecidos como opção ao comprar software de banco de dados e algumas funcionalidades são incorporadas ao banco de dados e não estão disponíveis separadamente na própria ferramenta ETL.
- Ferramenta ETL de Business Intelligence integrada: são produtos do mesmo fornecedor do software de BI, sendo, em muitos casos, produtos separados e que podem ser usados independente da ferramenta de BI.
- Ferramenta ETL de produto de nicho: são produtos que não se enquadram em nenhum dos grupos mencionados.

De acordo com Pall e Khaira (2013), algumas ferramentas de ETL se destacam:

- IBM InfoSphere Information Server oferece uma grande flexibilidade e é direcionada ao mercado como uma plataforma de metadados comum, tendo um alto nível de satisfação dos clientes e de fácil uso, e requer muito poder de processamento. Para maiores informações, acesse o website da IBM (IBM, 2019).
- Oracle Data Integrator é uma das ferramentas líderes de mercado de ETL, tendo uma estreita conexão com todos os aplicativos de *data warehousing* da Oracle e possui a tendência de integrar todas as ferramentas em uma aplicação e um ambiente. Para maiores informações, acesse o website da Oracle (ORACLE, 2019).
- Microsoft SQL Server Integration Services (SSIS) oferece facilidade e velocidade de implementação com integração de dados padronizada, recursos em tempo real, baseado em mensagens, de custo relativamente baixo e que fornecem um excelente modelo de suporte e distribuição. Para maiores informações, acesse o site do Microsoft SQL Server (MICROSOFT, 2019).

Segundo Pall e Khaira (2013), algumas outras ferramentas de ETL devem ser consideradas:

- Talend: é uma ferramenta de integração de dados *open-source*. Utiliza abordagem de geração de código e uma GUI, possuindo recursos de qualidade de dados.
- SAS Data Integrator: é uma ferramenta de integração de dados muito poderosa e com recursos de gerenciamento múltiplo, podendo trabalhar com vários sistemas operacionais e coletar dados de várias fontes.
- Pentaho: é uma ferramenta de BI *open-source* que possui um produto chamado Kettle para integração de dados e possui uma GUI de fácil uso. Possui um motor Java que processa os trabalhos e tarefas e move os dados entre muitos bancos de dados e arquivos diferentes.

- CloverETL: é uma ferramenta que oferece integração de dados e que pode transformar dados de forma fácil. É uma ferramenta visual que fornece controle total dos fluxos de dados e processos.

O Quadro 1 apresenta um comparativo das ferramentas ETL considerando alguns critérios para comparação, tais como plataforma, software como serviço, facilidade de uso, reusabilidade, *debugging*, correções de sintaxe e nomes de campos, compilação e validação, módulos separados, mecanismo de dados, tabelas unidas, conexões nativas e conexões em tempo real.

Quadro 1 – Comparativo das ferramentas ETL

Critério	IBM Information Services	Oracle Data Integrator	SQL Server Integration Services	Talend	SAS	Pentaho	Clover ETL
Comercia-lização	1996	1999	1997	2007	1996	2006	2005
Autônomo ou integrado	Autônomo	Autônomo	Autônomo	Autônomo	Autônomo	Autônomo	Autônomo
Plataforma	6	6	1	7	8	4	7
Versão	8.1	11.1.1.5	10	5.2	10	3.2	2.9.2
Baseado Engine ou código gerado	Ambos	Código gerado	Ambos	Código gerado	Código gerado	Baseado Engine	Baseado Engine
Software como serviço	Sim	Sim	-	Não	Não	Não autônomo	Não
Fácil uso	Sim	Sim	Sim	Sim	Sim	Não	Não
Reusabili-dade	Sim	Sim	Sim	Sim	Sim	Sim	Sim
Debugging	Sim	Sim	Sim	Sim	Sim	Não	Não
Correções de sintaxe e nome dos campos	Sim	Sim	-	Sim	Sim	Não	Sim
Compilar/ validar	Sim	Sim	Sim	Sim	Sim	Sim	Sim
Módulos separados	Não	Não	-	Sim	Sim	Não	Sim

Mecanismos de dados	*logging +triggers*	*message queuing +logging +triggers*	*message queuing +logging +triggers*	*message queuing +triggers*	*message queuing*	Não	*message queuing +triggers*
Juntar tabelas	Sim	Sim	Não	Sim	Sim	Não	Não
Informação de suporte	Sim	Sim	-	Sim	Sim	Sim	Sim
Conexões nativas	41	22	4	35	18	20	7
Conexões em tempo real	2	3	2	3	3	3	3

Fonte: PALL; KHAIRA, 2013.

Para o entendimento do comparativo apresentado no Quadro 1, alguns critérios foram utilizados:

- Plataforma: representa quantas plataformas suportam ETL.
- Software como serviço: esse critério verifica se o produto está disponível como software como serviço.
- Fácil de usar: inclui quão fácil de usar o produto se apresenta.
- Reusabilidade: se há componentes reutilizados e suporta funções definidas pelo usuário para estar disponível para outros programas.
- *Debugging:* se as ferramentas apresentam processo de *debug*.
- Correções de sintaxe e nomes de campos: algumas ferramentas não fornecem sugestão automática se há erros de sintaxe ou nomes de campos.
- Compilar e validar: verificar a facilidade de se localizar erros e, se houver, eles são destacados no código.
- Módulos separados: normalmente, a ferramenta é composta de pelo menos dois módulos, o módulo de tempo real e o módulo de lote. Em algumas ferramentas, eles podem ser adquiridos separadamente.
- Mecanismo de dados: os dados mudam quando são extraídos e transformados. O mecanismo de dados trata de como os dados alterados serão reconhecidos. Uso de *triggers* gatilhos), *logs* (registro) de entrada e *queuing* (fila de mensagens) são mecanismos.

- Tabelas unidas: verifica se é possível unir duas tabelas de maneira gráfica, permitindo que o banco de dados execute a união em vez de permitir que a ferramenta ETL junte as tabelas.
- Conexões nativas: quantas conexões nativas a ferramenta de ETL suporta.
- Conexões em tempo real: quantas conexões em tempo real a ferramenta pode conectar.

ASSIMILE

As empresas de *data warehousing* têm o dilema de escolher o processo de ETL certo e a ferramenta de ETL correta para a organização, pois um passo ou uma escolha errada pode levar a uma série de perdas, tanto monetariamente quanto pelo tempo, sem mencionar a quantidade de trabalho perdido. Escolher uma entre uma variedade de ferramentas de ETL deve ser feito com o conhecimento de suas características.

1.2 SQL Server Integration Services (SSIS)

De acordo com Cote, Lah e Sarka (2017), o SQL Server Integration Services (SSIS) é uma ferramenta que facilita a extração de dados, consolidação e opções de carregamento (ETL), apresentando aprimoramentos de codificação do SQL Server e *data warehousing*. Segundo Laudenschlager (2018), o Microsoft SQL Server Integration Services (SSIS) é uma plataforma utilizada para integração de dados em nível corporativo e é solução para transformação de dados, sendo usado para solucionar problemas empresariais complexos, realizando carregamento de *data warehouses*.

O SQL Server Integration Services (SSIS) permite extrair e transformar uma grande variedade de fonte de dados, tais como arquivos simples, arquivos de dados XML, fontes de dados relacionadas e transferir esses

dados para um ou mais destinos. O Integration Services apresenta um conjunto avançado de tarefas internas e ferramentas gráficas que são utilizadas para armazenamento, execução e gerenciamento de pacotes.

De acordo com a Devmedia (2014), a arquitetura do SSIS é formada por alguns componentes, tais como Solution (solução), Project (projeto), Package (pacote) Tasks (tarefas), etc., e pela hierarquia entre esses componentes, sendo o de maior hierarquia o Solution, que é formado por um ou mais Projects, que são formados por um ou mais Packages e que incluem uma ou mais Tasks, etc.

Solution, segundo o Devmedia (2014), é utilizado para gerenciar um ou mais projetos, reunindo todos os projetos que são usados para uma determinada solução de negócio. A solução pode ter diferentes tipos de projeto, como um projeto SSIS e um projeto SSAS (SQL – Server Analyses Services). A solução possui dois arquivos associados:

- Arquivo ".suo", contendo informações das preferências do desenvolvedor para utilizar a solução;
- Arquivo ".sln", contendo informações das configurações da solução e lista de projetos, sendo essa extensão usada para visualizar a solução.

Project, de acordo com Devmedia (2014), é o local onde são desenvolvidos os pacotes, podendo possuir vários pacotes, contendo três arquivos associados:

- Arquivo ".dtproj", com as configurações do projeto e lista de pacotes;
- Arquivo ".dtproj.user", com as informações de preferências do desenvolvedor para usar o projeto;
- Arquivo ".database", com as informações necessárias para o Business Intelligence Development Studio.

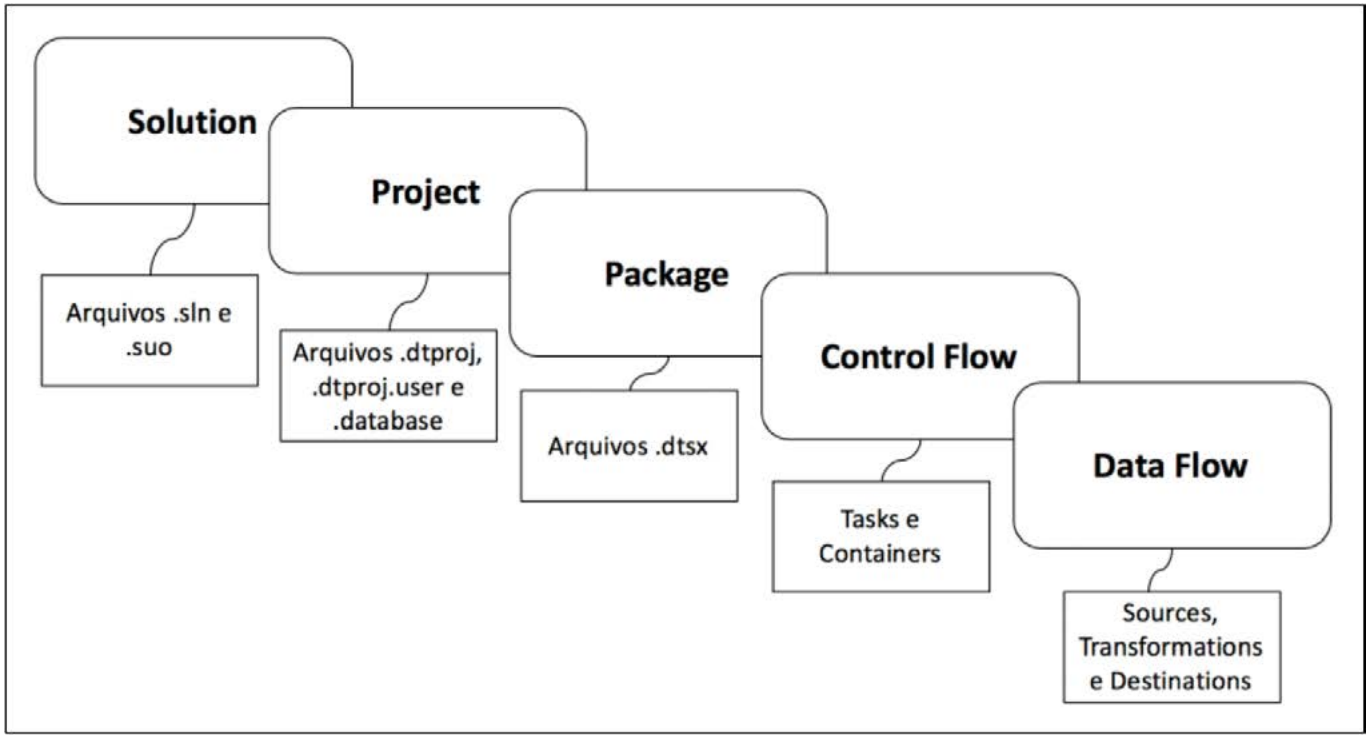
Package, segundo Devmedia (2014), é o local em que se desenvolvem os fluxos que podem ser Control Flow (controle) e Data Flow (dados). Ao criar um Package, ao menos um fluxo de controle deve ser adicionado e os pacotes devem ter a extensão ".dtsx".

Control Flow, de acordo com Devmedia (2014), são os principais componentes do pacote e seus principais itens são Containers e Tasks (tarefas). Containers são utilizados para agrupar e organizar as tarefas e as tarefas têm a função de executar alguma funcionalidade, como, por exemplo, executar instruções SQL, executar processos ou manipular arquivos do sistema. Por meio do fluxo de controle, é possível mover ou deletar arquivos, executar consultas ou procedures em banco de dados, consultar *web services*, realizar *backup* de banco de dados, etc. A principal tarefa é o Data Flow (fluxo de dados).

Data Flow, de acordo com Devmedia (2014), é uma tarefa do fluxo de controle usada quando se necessita realizar transferência de dados. O Data Flow (fluxo de dados) é utilizado para importar, exportar e transformar dados, ETL (Extract, Transform and Load).

A Figura 7 apresenta a relação de hierarquia que existe entre os componentes do SQL Server Integration Services (SSIS), em que o componente de maior nível é Solution e o de menor nível é o Data Flow.

Figura 7 – Hierarquia dos elementos do SSIS



Fonte: o autor.

A transferência pode ser de um arquivo de texto para um banco de dados, de um banco de dados para o Excel e entre bancos de dados diferentes (SQL Server e Oracle), etc. O fluxo de dados possui seus componentes: origem dos dados (Data Flow Sources), transformações (Data Flow Transformations) e destino dos dados (Data Flow Destinations).

De acordo com Knight et al. (2014), o que faz do SQL Server Integration Services (SSIS) tão importante são os recursos de movimentação e limpeza de dados, sendo uma ferramenta ETL, permitindo juntar fluxos de trabalhos complexos e fluxos de limpeza de dados.

Segundo Knight et al. (2014), no SQL Server 7.0, a Microsoft possuía uma pequena equipe de desenvolvedores atuando em um recurso muito discreto chamado Data Transformation Services (DTS), cujo objetivo era transformar dados de qualquer OLE DB para qualquer destino, porém o DTL não possuía alguns componentes necessários para o suporte de processos ETL.

Com o SQL Server 2005, a Microsoft incluiu o SSIS, que não era mais um recurso discreto como o DTS, sendo realizado um enorme investimento em usabilidade, com aprimoramentos na caixa de ferramentas e gerenciamento e implantação do SSIS.

Dessa forma, você aprendeu as definições e os conceitos básicos dos *data warehouses*, bem como os processos de extração, transformação e carga (ETL) e arquiteturas de *data warehousing*.

TEORIA EM PRÁTICA

Uma empresa nacional de revenda de cosméticos está enfrentando alguns problemas financeiros. Dado o grande volume de produtos lançados pelo setor de cosméticos,

a empresa está tendo dificuldades em acompanhar a demanda de lançamentos, o que, muitas vezes, acaba por comprometer o capital de giro investido em produtos com baixa procura. Outro problema são os produtos que possuem prazo de validade curto, que acabam por vencer e não podem ser trocados pelo fornecedor. Hoje a empresa conta com um sistema de vendas e controle de estoque, com banco de dados relacional e um processo de marketing pelas redes sociais, porém está tendo dificuldades para a tomada de decisões relacionadas ao que o seu público-alvo realmente consome e assim evitar gastos desnecessários. Para que a empresa possa construir um *data warehouse* e utilizar posteriormente um sistema de Business Intelligence, como você poderia auxiliar a empresa na obtenção de dados limpos e que possam ser usados para a tomada de decisões?

VERIFICAÇÃO DE LEITURA

1. O processo de integração de dados envolve práticas, técnicas arquiteturais e ferramentas para obter acesso consistente e para entregar dados em uma grande área de assuntos e tipos de estrutura em uma empresa. Assinale a alternativa que apresente, corretamente, as categorias das ferramentas de ETL:

   a. ETL codificado manualmente e ETL baseado em ferramenta.

   b. ETL automatizado e ETL baseado em ferramenta.

   c. ETL codificado manualmente e ETL automatizado.

d. ETL codificado e ETL baseado em ferramenta.

e. ETL codificado manualmente e ETL codificado.

2. São produtos independentes da base de dados e ferramenta de Business Intelligence que se pretende usar. As empresas não precisam depender de nenhum outro produto para a funcionalidade oferecida e permitem a migração para diferentes bases de dados sem mudar o processo de integração. Assinale a alternativa que apresenta, corretamente, a categoria de classificação da ferramenta de ETL descrita acima:

   a. Ferramentas ETL base de dados integrada.

   b. Ferramentas ETL pura.

   c. Ferramentas ETL Business Intelligence Integrated.

   d. Ferramenta ETL de produto de nicho.

   e. Ferramenta ETL baseado em ferramenta.

3. É utilizado para gerenciar um ou mais projetos, reunindo todos os projetos que são usados para uma determinada solução de negócio. Assinale a alternativa que apresenta, corretamente, a arquitetura descrita.

   a. Project.

   b. Package.

   c. Solution.

   d. Control Flow.

   e. Data Flow.

Referências bibliográficas

COTE, C.; LAH, M., SARKA, D. **SQL Server 2017 Integration Services Cookbook:** ETL techniques to load and transform data from various sources using SQL Server 2017 Integration Services. Birmingham: Packt Publishing, 2017.

DEVMEDIA. **Microsoft ETL:** Arquitetura SSIS. 2014. Disponível em: https://www.devmedia.com.br/microsoft-etl-arquitetura-ssis-sql-server-integration-services/30862. Acesso em: 18 mar. 2019.

IBM. **IBM InfoSphere Information Server**. Disponível em: https://www.ibm.com/us-en/marketplace/infosphere-information-server. Acesso em: 18 mar. 2019.

KNIGHT, B. et al. **Professional Microsoft SQL Server 2014 Integration Services**. Indianapolis: John Wiley & Sons Inc., 2014.

LAUDENSCHLAGER, D. **SQL Server Integration Services**. 2018. Disponível em: https://docs.microsoft.com/pt-br/sql/integration-services/sql-server-integration-services?view=sql-server-2017. Acesso em: 18 mar. 2019.

MICROSOFT. Microsoft SQL Server 2017. Disponível em: https://www.microsoft.com/pt-br/sql-server/sql-server-downloads. Acesso em: 18 mar. 2019.

ORACLE. **Oracle Data Integrator**. Disponível em: https://www.oracle.com/middleware/technologies/data-integrator.html. Acesso em: 18 mar. 2019.

PALL, A. S.; KHAIRA, J. S. A comparative review of Extraction, Transformation and Loading Tools. **Database Systems Journal**, Jalandhar, v. IV, n. 2, p. 42-51, 14 fev. 2013.

TURBAN, E. et al. **Business Intelligence:** um enfoque gerencial para a inteligência do negócio. Porto Alegre: Bookman, 2009.

Gabarito

Questão 1 – Resposta A

As categorias das ferramentas de ETL são ETL codificado manualmente e ETL baseado em ferramenta.

Questão 2 – Resposta B

Ferramenta de ETL pura são produtos independentes da base de dados e ferramenta ETL *Business Intelligence* que se pretende usar. As empresas não precisam depender de nenhum outro produto para a funcionalidade oferecida e permitem a migração para diferentes bases de dados sem mudar o processo de integração.

Questão 3 – Resposta C

Solution é utilizado para gerenciar um ou mais projetos, reunindo todos os projetos que são usados para uma determinada solução de negócio.

Extração de dados

Autor: Thiago Salhab Alves

Objetivos

- Compreender as definições e os conceitos básicos de extração de dados.
- Aprender a criar um mapa de dados lógico, que documente a relação entre os campos de origem e os campos de destino da tabela.
- Conhecer os diferentes tipos de fontes de dados de origem que devem ser extraídos.

1. Extração de dados

Prezado aluno, você já parou para pensar como as organizações, nacionais e multinacionais, realizam o processo de extração de dados para compor um *data warehouse*? Diversas são as fontes de dados, que apresentam dados heterogêneos e que necessitam ser combinados para compor uma base de dados unificada. Como adotar um processo de extração para essas diversas fontes de dados de origem? Dessa forma, faz-se necessário conhecer o processo de extração de dados, criar um mapa de dados lógico e os diferentes tipos de dados de origem que devem ser extraídos.

Esta leitura irá apresentar as definições e os conceitos básicos de extração de dados. Você aprenderá a criar um mapa de dados lógico, que documente a relação entre os campos de origem e os campos de destino da tabela e conhecer os diferentes tipos de fontes de dados de origem que devem ser extraídos. Uma boa aula e bom trabalho.

1.1 Processo de extração de dados

De acordo com Kimball e Caserta (2009), o primeiro passo da integração é extrair com sucesso dados dos principais sistemas de origem. Cada fonte de dados possui um conjunto distinto de características que precisam ser gerenciadas para extrair de forma efetiva os dados para o processo de ETL. As empresas evoluem e adquirem ou herdam vários sistemas computadorizados, com a finalidade de ajudar na administração de seus negócios, tais como sistemas de ponto de venda, gerenciamento de inventário, controle de produção e de contabilidade geral, mas que são frequentemente física e logicamente incompatíveis.

Segundo Elias (2014), a etapa de extração pode ser entendida como a fase em que os dados são extraídos de diversas fontes organizacionais e conduzidos para uma área de transição em que os dados são convertidos pra um único formato. A conversão se faz necessária

devido à heterogeneidade existente nas informações provenientes de várias fontes, sendo importante uma conformação prévia para o tratamento adequado.

De acordo com Ribeiro, Cavalcanti e Goldschmidt (2009), na etapa de extração, os dados são capturados de múltiplas fontes, sendo necessário diferentes ferramentas adaptadas para cada fonte.

O processo ETL necessita integrar efetivamente sistemas diferentes:

- Sistemas gerenciadores de banco de dados;
- Sistemas operacionais;
- Hardware;
- Protocolos de comunicação.

De acordo com Kimball e Caserta (2009), antes de construir um sistema de extração, é necessário criar um mapa de dados lógico, que documente a relação entre os campos de origem e os campos de destino da tabela. A implementação pode fracassar se não for cuidadosamente projetada antes de ser implementada. Antes de iniciar qualquer desenvolvimento de ETL físico, certifique-se de que as etapas seguintes são atendidas:

- Tenha um plano: o processo ETL deve ser projetado logicamente e documentado. O mapa lógico de dados é fornecido ao engenheiro do *data warehouse* e é a especificação para a equipe de ETL para criar o ETL físico. Esse documento é chamado de relatório de linhagem de dados e é a base dos metadados que são usados pelos testadores para garantir a qualidade e para os usuários finais descreverem exatamente o que é feito entre o sistema de origem e o *data warehouse*.
- Identifique fontes de dados candidatas: identifique as prováveis fontes de dados que acredite apoiar as decisões necessárias, verificando nessas fontes de dados elementos que sejam úteis para os dados do usuário final, sendo esses elementos de dados a entrada para a criação do perfil de dados.

- Analise sistemas de origem com uma ferramenta de criação de perfil de dados: os dados nos sistemas de origem devem ser examinados quanto a qualidade, integridade e adequação para seu propósito. A etapa de criação do perfil de dados deve ser feita por alguém que tenha a habilidade de identificar quais dados serão usados no *data warehouse* para as necessidades de tomada de decisões. Os dados em qualquer sistema de origem devem ser analisados e os melhores esforços devem ser realizados para aplicar regras de negócios para corrigir os dados antes que sejam carregados no *data warehouse*.
- Receba as instruções para a linhagem dos dados e regras de negócios: depois que os dados foram qualificados pela etapa de criação de perfil de dados e o modelo de dados de destino for entendido, o engenheiro do *data warehouse* e os analistas de negócios devem orientar os desenvolvedores de ETL por meio da linhagem de dados e regras de negócios para extrair, transformar e carregar as áreas de assunto no *data warehouse*. A etapa de criação do perfil de dados deve ter criado duas subcategorias de regras de negócios específicas de ETL:
    - O alterações necessárias nos dados durante as etapas de limpeza de dados;
    - O coerções a atributos dimensionais e fatos numéricos medidos para alcançar a conformidade padrão entre fontes de dados separadas.
- Receba as instruções do modelo de dados do *data warehouse*: a equipe de ETL deve entender completamente o modelo de dados físicos do *data warehouse*, incluindo o entendimento dos conceitos de modelagem dimensional. A equipe de desenvolvimento deve ter um entendimento completo de como dimensões, fatos e outras tabelas no modelo dimensional funcionam juntas para implementar soluções de ETL.

- Valide cálculos e fórmulas: verifique com os usuários finais quaisquer cálculos especificados na linha de dados, evitando, assim, que se implementem medidas erradas no *data warehouse*, certificando-se de que os cálculos estejam corretos antes de codificar algoritmos errados no processo de ETL.

Segundo Kimball e Caserta (2009), antes de se conhecerem os detalhes das várias fontes de dados que serão extraídos, é necessário conhecer o documento de mapeamento de dados lógicos. Esse mapeamento é apresentado na Tabela 1 com os seguintes elementos:

- Nome da tabela de destino: o nome físico da tabela conforme utilizado no *data warehouse*.
- Nome da coluna de destino: o nome da coluna na tabela do *data warehouse*.
- Tipo de tabela: indica se a tabela é um fato, dimensão ou subdimensão.
- Banco de dados de origem: o nome da instância do banco de dados em que os dados de origem residem, sendo geralmente uma *string* de conexão necessária para se conectar ao banco de dados.
- Nome da tabela de origem: o nome da tabela em que os dados se originam. Pode haver o caso de se usar mais de uma tabela, bastando listar todas as tabelas necessárias para preencher a tabela no *data warehouse* de destino.
- Nome da coluna de origem: a coluna ou colunas necessárias para preencher o destino, bastando listar todas as colunas necessárias para carregar a coluna de destino.
- Transformação: manipulação dos dados de origem para que correspondam ao formato esperado no destino, sendo um componente anotado em SQL.

Tabela 1 – Mapa de dados lógicos

Dados destino				Dados fonte			
Nome Tabela	Nome Coluna	Tipo de dado	Nome banco	Nome tabela	Nome coluna	Tipo de dado	Transfor-mação
EMPLOYEE_DIM	EMPLOYEE_ID	NUMBER	HR_SYS	EMPLOYEES	EMPLOYEE_ID	NUMBER	Chave para Employee no sistema HR
EMPLOYEE_DIM	BIRTH_ COUNTRY_NAME	VARCHAR (75)	HR_SYS	COUNTRIES	NAME	VARCHAR (75)	select c.name from employees e, states s, countries c where e.state_id = s.state_id and s.country_id = c.country
EMPLOYEE_DIM	BIRTH_STATE	VARCHAR (75)	HR_SYS	STATES	DESCRIPTION	VARCHAR (255)	select s.description from employees e, states s where e.state_id = s.state_id
EMPLOYEE_DIM	DISPLAY_NAME	VARCHAR (75)	HR_SYS	EMPLOYEES	FIRST_NAME	VARCHAR (75)	select initcap (salutation) ||' '||initcap (first_name) ||' '|| initcap(last_ name) from employee

Fonte: adaptado de Kimball e Caserta (2009, p. 60).

Segundo Kimball e Caserta (2009), de acordo com a Tabela 1, os tipos de dados entre origem e destino para STATES são convertidos de 255 caracteres para 75 caracteres, perdendo potencialmente os dados.

Algumas ferramentas de ETL abortariam ou falhariam em todo o processo com esse tipo de erro de estouro de dados. Observe que, na notação de transformação para STATES, não se define explicitamente essa conversão de dados, sendo de responsabilidade da equipe de ETL assumir a responsabilidade de lidar explicitamente com esses tipos conversões implícitas.

Segundo Kimball e Caserta (2009), o sucesso do *data warehouse* é a limpeza e coesão dos dados nele contidos. Um armazenamento de dados unificado exige uma visão completa de cada um dos seus sistemas de dados de origem, sendo necessário incluir a etapa de entendimento dos dados no projeto de ETL. A análise do sistema de origem se divide em duas fases: a fase de descoberta de dados e a fase de detecção de anomalias nos dados.

Na fase de descoberta de dados, a equipe de ETL deve se aprofundar mais na descoberta dos dados para determinar cada sistema, tabela e atributo de origem necessário para carregar o *data warehouse*. Deve-se determinar a fonte adequada para cada elemento, em que uma boa análise evita atrasos causados pelo uso de uma fonte errada.

De acordo com Kimball e Caserta (2009), o processo de análise dos sistemas de origem é utilizado para uma melhor compreensão do seu conteúdo. Esse entendimento é obtido pela aquisição dos diagramas entidade-relacionamento (DER). Caso o diagrama entidade-relacionamento não exista, pode ser criado, utilizando para isso o processo de engenharia reversa, que é uma técnica que desenvolve um diagrama entidade-relacionamento a partir dos metadados do banco de dados existente.

Segundo Kimball e Caserta (2009), podem-se encontrar diferentes fontes de dados que necessitam ser usadas no *data warehouse*, levando ao processo de integração dessas diferentes fontes. Integrar dados significa muito mais do que simplesmente coletar fontes de dados distintas e armazenar esses dados em um único repositório. São atividades para integração de dados:

- Identificar os sistemas de origem: durante a fase de criação do perfil de dados para construção do mapeamento de dados lógicos, a equipe do *data warehouse* deve trabalhar em conjunto para detectar as potenciais fontes de dados para o *data warehouse* e nomear um sistema de registro para cada elemento.

- Compreender os sistemas de origem (criação do perfil de dados): após identificar os sistemas de origem, é necessário realizar uma análise completa de cada sistema, que faz parte da criação do perfil de dados. A análise visa encontrar anomalias e problema de qualidade de dados.
- Criar e registar a lógica de correspondência: após o entendimento de todos os atributos das entidades, o próximo passo é projetar um algoritmo de correspondência para permitir que entidades nos sistemas diferentes sejam unidas, às vezes bastando identificar a chave primária das várias tabelas de clientes.
- Estabelecer as regras de negócio de atributos não chave: os dados são provenientes de várias tabelas e colunas do sistema que normalmente contêm muitos atributos diferentes. Considere, por exemplo, uma lista de departamentos com dados vindos da base do departamento de RH, porém o código contábil do departamento vem do sistema financeiro. Embora o sistema de RH seja o sistema de registro, alguns atributos podem ser mais confiáveis em outros sistemas. Assim, atribuir regras de negócios para atributos não chave é importante quando os atributos existem em vários sistemas.
- Carregar dimensão conformada: a tarefa final do processo de integração é carregar fisicamente o dado conformado.

De acordo com Kimball e Caserta (2009), cada fonte de dados pode estar em um Sistema Gerenciador de Banco de Dados (SGBD) diferente e em uma plataforma diferente. Em um projeto de *data warehouse*, pode haver a necessidade de se comunicar com sistemas de diferentes origens. O ODBC (Open Database Connectivity) foi criado para permitir que os usuários acessassem bancos de dados a partir de seus aplicativos para Windows, tendo como intenção tornar os aplicativos portáteis e, caso algum banco de dados de um aplicativo fosse alterado, por exemplo, de DB2 para Oracle, a camada de aplicativo não precisaria ser recodificada e compilada para acomodar a alteração, bastando alterar o driver ODBC.

Segundo Kimball e Caserta (2009), em muitas empresas de grande porte, muitos dados corporativos do dia a dia são processados e armazenados em *mainframes* e integrar dados desses sistemas no *data warehouse* envolve alguns desafios. Existem algumas características dos sistemas de *mainframe* que o ETL deve saber lidar:

- Cobol Copybooks: Cobol continua sendo uma linguagem de programação dominante em *mainframes* e o *layout* de arquivo para dados é descrito em Cobol Copybooks. Ele define os nomes de campos e tipos de dados associados para um arquivo de dados de *mainframe*. Como acontece com outros arquivos simples que se encontram no processo ETL, apenas dois tipos de dados existem em arquivos simples de *mainframes*: texto e numérico, e os valores numéricos são armazenados de várias maneiras que se necessitam conhecer para processar com precisão. Tipos de dados de texto e numéricos são indicados no Cobol Copybooks com a cláusula PIC. PIC X denota campo de texto, e PIC 9, que o campo é numérico.

- Conjunto de Caracteres EBCDIC: tanto em sistemas de *mainframe* quanto em baseados em UNIX e Windows, os dados são armazenados como bits e bytes, de modo que os dados do sistema *mainframe* podem ser prontamente utilizados em sistemas UNIX e Windows. Contudo, os sistemas UNIX e Windows usam o conjunto de caracteres ASCII (American Standard Code for Information Interchange[1]) enquanto os *mainframes* usam um conjunto diferente, conhecido como EBCDIC (Extended Binary Coded Decimal Interchange Code[2]). O EBCDIC usa mais ou menos os mesmos caracteres que o ASCII, mas usa diferentes combinações de 8 bits para representá-los. Por exemplo, a letra a nas tabelas ASCII é representado para caractere 97 (01100001), mas em EBCDIC, o caractere 97 é /. Para usar dados de *mainframes* em sistemas UNIX ou Windows, deve ser traduzido de EBCDIC para ASCII, que é traduzido de forma bastante simples e automática, utilizando, por exemplo, o FTP (File Transfer Protocol), realizando a tradução.

[1] Código Padrão Americano para Intercâmbio de Informações.

[2] Código Binário de Intercâmbio Decimal Codificado Estendido.

- Dados numéricos: para diferenciar valores numéricos, com pontos decimais, são utilizados no Cobol Copybooks o PIC. Assim, é possível diferenciar o valor 25.000.01 que é armazenado como 002500001 de 2.500.001 que é armazenado da mesma maneira. A cláusula PIC pode dar ao mesmo valor de dados significados diferentes. Para processar com precisão um valor numérico proveniente de um *mainframe*, deve-se primeiro transformá-lo em seu formato de exibição antes de transmiti-lo ao sistema de *data warehouse*. Por exemplo, PIC 9(9) identifica que o dado 002500001 representa o valor 2,500,001 e PIC 9(7)V99 identifica que o dado 002500001 representa o valor 25,000.01.

De acordo com Kimball e Caserta (2009), arquivos simples são a base de qualquer aplicativo de armazenamento de dados. Arquivos simples são usados pelo processo de ETL por pelo menos três razões:

- Entrega dos dados de origem: quando dados são originais de fontes como *mainframes* e usam o FTP para a área de armazenamento de dados em arquivos simples. Os dados provenientes de bancos de dados ou planilhas são geralmente fornecidos por meio de arquivos simples.

- Tabelas de trabalho: as tabelas de trabalho são criadas pelo processo ETL para seu uso. Na maioria das vezes, os arquivos simples são usados porque as leituras e gravações no sistema de arquivos são muito mais rápidas do que a inserção e a consulta de um SGBD.

Segundo Kimball e Caserta (2009), outras fontes de dados para um *data warehouse* estão no padrão XML (Extensible Markup Language[3]), que surgiu para se tornar uma linguagem natural para troca de dados entre empresas. Para se processar XML, deve-se entender como ele funciona. O XML tem dois elementos importantes: seus metadados e os dados. Para processar um documento XML, deve-se conhecer a estrutura do documento. A estrutura de um documento XML é geralmente fornecida em um arquivo separado, contendo um DTD ou um XML Schema:

[3] Linguagem de Marcação Extensível.

- DTD (Definição de Tipo de Documento): arquivo que descreve a estrutura de dados no documento ou arquivo XML.
- XML Schema: é o sucesso do DTD e são mais ricos e úteis que o DTD. Um XML Schema permite que uma instrução SQL CREATE TABLE seja definida diretamente.

De acordo com Kimball e Caserta (2009), a fonte de log da web está presente nas empresas que possuem um site. Cada site possui registros da web que armazenam todos os objetos publicados ou veiculados do servidor da web. Os logs da web são importantes porque revelam o tráfego dos usuários no site. Entender o comportamento dos usuários em seu site é muito valioso e os registros da web fornecem essa informação. A atividade de analisar logs da web e armazenar os resultados em um data mart para análise é conhecido como *data warehousing* do *clickstream*.

Segundo Kimball e Caserta (2009), os sistemas de planejamento de recursos empresariais (ERP) foram criados para solucionar um dos problemas enfrentados pelos *data warehouses*, que é a integração de dados heterogêneos. Os ERPs são projetados para serem uma solução corporativa integrada que permite que todas as principais entidades da empresa, como vendas, contabilidade, recursos humanos, inventário e controle de produção, estejam na mesma plataforma.

Os sistemas ERPs são grandes e seus modelos de dados são abrangentes, muitas vezes contendo milhares de tabelas. Muitos fornecedores de ERP, dada a sua complexidade, fornecem adaptadores ERP para se comunicar com os populares sistemas ERP, ajudando a navegar pelos metadados dos sistemas e entendimento da aplicação.

De acordo com Kimball e Caserta (2009), ao realizar o carregamento inicial, as alterações de conteúdo de dados nos dados de origem não são tão importantes porque está sendo extraída a fonte de dados completa. Quando a extração estiver completa, capturar alterações de dados no

sistema de origem se torna uma necessidade. Esperar que a carga inicial seja completada para planejar as técnicas de captura de dados alterados irá gerar vários problemas, devendo ser planejada uma estratégia de captura de alterações nos dados de origem no início do projeto.

Segundo Kimball e Caserta (2009), há várias maneiras de se capturar alterações nos dados de origem. O uso de colunas de auditorias que são anexadas ao final de cada tabela para armazenar a data e a hora em que um registro foi adicionado ou modificado. Essas colunas são preenchidas por meio de acionadores de banco de dados disparados automaticamente à medida que os registros são inseridos ou atualizados (*trigger*).

ASSIMILE

Segundo Kimball e Caserta (2009), uma situação muito comum que impede o processo de ETL de usar colunas de auditoria é quando os campos são preenchidos pelo aplicativo *front-end* e a equipe de DBA permite que os scripts de *back-end* modifiquem os dados. Dessa forma, enfrenta-se um alto risco de eventualmente perder os dados alterados durante as cargas incrementais. Para minimizar seu risco, faça com que todos os scripts de *back-end* sejam validados por uma equipe de garantia de qualidade que insista e teste se os campos de autoria são preenchidos pelo script antes de ser aprovado.

Considere os seguintes pontos por Kimball e Caserta (2009) sobre o processo de extração:

- Restringir colunas indexadas: atuar junto ao DBA para garantir que todas as colunas na sua cláusula WHERE sejam indexadas no sistema de origem, caso contrário, provocará uma varredura da base de dados inteira.

- Recupere os dados de que necessita: uma boa consulta retorna exatamente o que você precisa. Não se deve recuperar uma tabela inteira e filtrar dados indesejados posteriormente na ferramenta de ETL. Uma exceção é quando o DBA do sistema de transação se recusar a indexar as colunas necessárias para restringir as linhas retornadas na consulta ou quando é necessário baixar todo o banco de dados de origem.
- Utilize a cláusula DISTINCT com moderação: a cláusula DISTINCT é lenta e encontrar um equilíbrio entre a execução de um DISTINCT durante a consulta de extração em vez de agregar ou agrupar os resultados em sua ferramenta de ETL é desafiador e depende da porcentagem de duplicatas na origem.
- Utilize o operador SET com moderação: UNION, MINUS E INTERSECT são operadores SET e, assim como o DISTINCT, são lentos, mas em algumas situações, esses operadores não podem ser evitados. Utilize UNION ALL em vez de UNION.
- Utilize HINT conforme necessário: a maioria dos bancos de dados suporta a palavra-chave HINT e pode ser usado, porém é mais imporante forçar sua consulta a usar um índice específico.
- Evite NOT: se possível, evite restrições e junções NOT. Independente de usar a palavra-chave NOT ou os operadores <>, o banco de dados provavelmente irá optar por digitar uma tabela completa em vez de usar índices.
- Evite funções em sua cláusula WHERE: difícil de se evitar, especialmente quando se restringe as datas. Experimente diferentes técnicas antes de se comprometer com o uso de uma função em sua cláusula WHERE.

O objetivo da consulta de extração, segundo Kimball e Caserta (2009), é obter todas as chaves relevantes. Pode ser tão simples quanto selecionar várias colunas, quanto selecionar várias colunas de uma tabela, ou ser tão completo quanto criar dados inexistentes e pode variar de ter de unir algumas tabelas ou unir várias tabelas de fontes de dados heterogêneos.

PARA SABER MAIS

De acordo com Kimball e Caserta (2009), o registro de fatos excluídos ou sobrescritos dos sistemas de origem podem representar um desafio muito difícil para o *data warehouse* se nenhuma notificação de exclusão ou substituição ocorre. Para isso, siga os seguintes procedimentos:

- Negociar com os proprietários do sistema de origem, se possível, notificação explícita de todos os registros excluídos ou substituídos.
- Verificar constantemente o histórico do sistema para alertar a equipe ETL que algo mudou.

Dessa forma, você aprendeu as definições e os conceitos básicos de extração de dados, bem como a criar um mapa de dados lógico para documentar a relação entre campos de origem e campos de destino e conhecer os diferentes tipos de fontes de dados de origem que devem ser extraídos.

TEORIA EM PRÁTICA

Uma empresa nacional de revenda de cosméticos está enfrentando alguns problemas financeiros. Dado o grande volume de produtos lançados pelo setor de cosméticos, a empresa está tendo dificuldades em acompanhar a demanda por produtos lançados, o que muitas vezes acaba por comprometer o resultado financeiro, por investir em produtos com baixa procura. Outro problema são os produtos que possuem prazo de validade curto, que acabam por vencer e não podem ser trocados pelos

fornecedores. Hoje, a empresa conta com um sistema de vendas e controle de estoque, com banco de dados relacional e um processo de marketing pelas redes sociais, porém está tendo dificuldades para a tomada de decisões relacionadas ao que o seu público-alvo realmente consome e, assim, evitar gastos desnecessários. Como podemos organizar um processo de extração de dados do sistema de vendas e controle de dados e do marketing das redes social para poder criar um *data warehouse*?

VERIFICAÇÃO DE LEITURA

1. Antes de se construir um sistema de extração, é necessário criar um documento que apresente a relação entre os campos de origem e os campos de destino da tabela. Assinale a alternativa que apresente corretamente o documento ao qual o texto se refere.

   a. Mapa de dados lógico.

   b. Mapa de extração de dados.

   c. Mapa de extração.

   d. Mapa de extração lógico.

   e. Mapa lógico.

2. Durante a fase de criação do perfil de dados para construção do mapeamento de dados lógicos, a equipe do *data warehouse* deve trabalhar em conjunto para detectar as potenciais fontes de dados para o *data*

warehouse e nomear um sistema de registro para cada elemento. Assinale a alternativa que apresenta, corretamente, o processo descrito acima.

a. Identificar os sistemas de origem.

b. Compreender os sistemas de origem.

c. Criar e registrar a lógica de correspondência.

d. Estabelecer as regras de negócio de atributos não chave.

e. Carregar dimensão conformada.

3. Assinale a alternativa que apresente corretamente o primeiro passo do processo de integração de dados.

a. Limpeza de dados.

b. Extração de dados.

c. Conformidade de dados.

d. Transformação de dados.

e. Carregamento de dados.

Referências bibliográficas

ELIAS, D. **Entendendo o processo de ETL**. 2014. Disponível em: https://canaltech.com.br/business-intelligence/entendendo-o-processo-de-etl-22850/. Acesso em: 9 mai. 2019.

KIMBALL, R.; CASERTA, J. **The Data Warehouse ETL Toolkit:** practical techniques for extracting, cleaning, conforming, and data delivering data. Indianopolis: Wiley Publishing, 2009.

RIBEIRO, L. S.; CAVALCANTI, M. C; GOLDSCHMIDT, R. R. Complementação dos dados no contexto do processo de ETL. In: XXIV Simpósio Brasileiro de Banco de Dados – Workshop de Teses e Dissertações de Banco de Dados, 2009. **Anais...**. Fortaleza, 2009. p. 55-60.

Gabarito

Questão 1 – Resposta A

Antes de se construir um sistema de extração, é necessário criar um mapa de dados lógico, um documento que apresenta a relação entre os campos de origem e os campos de destino da tabela.

Questão 2 – Resposta A

Identificar os sistemas de origem é realizado durante a fase de criação do perfil de dados para construção do mapeamento de dados lógicos; a equipe do *data warehouse* deve trabalhar em conjunto para detectar as potenciais fontes de dados para o *data warehouse* e nomear um sistema de registro para cada elemento.

Questão 3 – Resposta B

O primeiro passo no processo de integração dos dados é a extração dos dados.

Limpeza de dados

Autor: Thiago Salhab Alves

Objetivos

- Compreender as definições e os conceitos de limpeza de dados.
- Aprender a remover “ruídos” e dados irrelevantes, como também corrigir inconsistências nos dados.
- Aprender as técnicas de remoção de “ruídos”, conhecidas como compartimentalização (*binning*), regressão e agrupamento (*clustering*).

1. Limpeza de dados

Prezado aluno, você já parou para pensar como as organizações nacionais e multinacionais realizam o processo de limpeza dos dados após a extração de dados para compor um *data warehouse*? Diversas são as fontes de dados que apresentam dados heterogêneos e que necessitam ser combinados para compor uma base de dados unificada. Como adotar um processo de limpeza para essas diversas fontes de dados de origem? Dessa forma, faz-se necessário conhecer o processo de limpeza de dados.

Esta leitura irá apresentar as definições e os conceitos de limpeza de dados. Você aprenderá a remover "ruídos", dados irrelevantes e a corrigir inconsistências nos dados, além de conhecer as técnicas de remoção de "ruídos", conhecidas como compartimentalização (*binning*), regressão e agrupamento (*clustering*). Uma boa aula e bom trabalho.

1.1 Processo de limpeza de dados

De acordo com Han e Kamber (2006), muitas vezes os dados estão em grandes quantidades, apresentando dificuldades de manipulação, e nem todos os dados são necessários no processo de ETL. Os dados se originam de diferentes fontes, quase sempre heterogêneas, possuindo:

- Formatos diferentes;
- Campos nem sempre preenchidos;
- Campos duplicados.

Considere como exemplo uma financeira que necessita obter informações sobre potenciais clientes. Para isso, comprou bases de dados de diferentes instituições e caberá a ela consolidar as seguintes bases de dados adquiridas:

- Bancos;
- Serviços de proteção ao crédito;
- Lojas vinculadas à associação de lojistas da cidade.

A estrutura da base recebida do banco apresentava os seguintes campos:

- CPF;
- Endereço;
- Limite no cartão;
- Limite cheque especial;
- Limite consignado;
- Se possui empréstimos;
- Se atrasou pagamentos;
- Se possui investimentos.

A estrutura da base recebida de uma loja de lingerie apresentava os seguintes campos:

- CPF;
- Endereço;
- Se usa cartão;
- Se parcela compras;
- Se atrasou pagamentos;
- Busto;
- Quadril;
- Cintura.

A estrutura da base recebida do Serviço de Proteção ao Crédito apresentava os seguintes campos:

- CPF;
- Nome;
- Débitos pendentes;
- Situação das pendências.

Com base nesses campos, como consolidar esta base de dados? Simplesmente unir os campos? Provavelmente haverá endereços duplicados e possivelmente inconsistentes e nem todos os clientes compraram na loja de lingerie.

Alguns campos podem estar vazios e todos os campos são relevantes? Tamanho de busto, cintura e quadril interferem na análise de crédito?

Em aplicações práticas, quase sempre será necessário realizar um pré-processamento dos dados para posterior utilização.

A limpeza de dados, segundo Han e Kamber (2006), é utilizada para remover "ruídos", dados irrelevantes e corrigir inconsistências nos dados. As rotinas de limpeza dos dados tradicionalmente buscam:

- Preencher valores faltantes;
- Suavizar os dados, eliminando ruído e detectando *outliers* (dados que apresentam grande afastamento dos demais dados da série);
- Corrigir inconsistências presentes nos dados.

Segundo Kimball e Caserta (2009), dados precisos apresentam as seguintes características:

- Corretos: os valores e descrições nos dados descrevem seus objetos de verdade e devem estar corretos; por exemplo, uma pessoa vive em uma cidade chamada São Pedro; dados corretos sobre esse endereço devem conter São Pedro como o nome correto da cidade.
- Não ambíguos: os valores e descrições nos dados devem ter apenas um significado; por exemplo, existem pelo menos dez municípios no Brasil com "São Pedro" no nome, mas há apenas um município no estado de São Paulo que se denomina "São Pedro", com esse nome exato; portanto, dados precisos sobre um endereço neste município precisam conter São Pedro como nome da cidade e São Paulo como o nome do estado para não ser ambíguo.

- Consistente: os valores e descrições nos dados usam uma notação para transmitir seus dados; por exemplo, no Brasil, o estado de São Paulo pode ser expresso em dados como SP ou São Paulo. Para ser consistente, dados precisos sobre o endereço devem utilizar apenas uma convenção (como São Paulo) para nomes de estados.

- Completo: o primeiro aspecto é garantir que os valores e descrições individuais nos dados estejam definidos (não nulos) para cada instância, por exemplo, garantindo que todos os registros tenham os endereços atuais; o segundo aspecto garante que o número de registros esteja completo ou garante que não se perderam registros.

De acordo com Kimball e Caserta (2009), quatro prioridades inter-relacionadas moldam os objetivos de qualidade dos dados:

- Completude: o subsistema de limpeza de dados necessita ser minucioso em sua detecção, correção e documentação da qualidade das informações que publica; os usuários finais querem utilizar o *data warehouse* como uma fonte de dados confiável, uma base sobre a qual podem construir suas métricas, estratégias e políticas de gerenciamento.

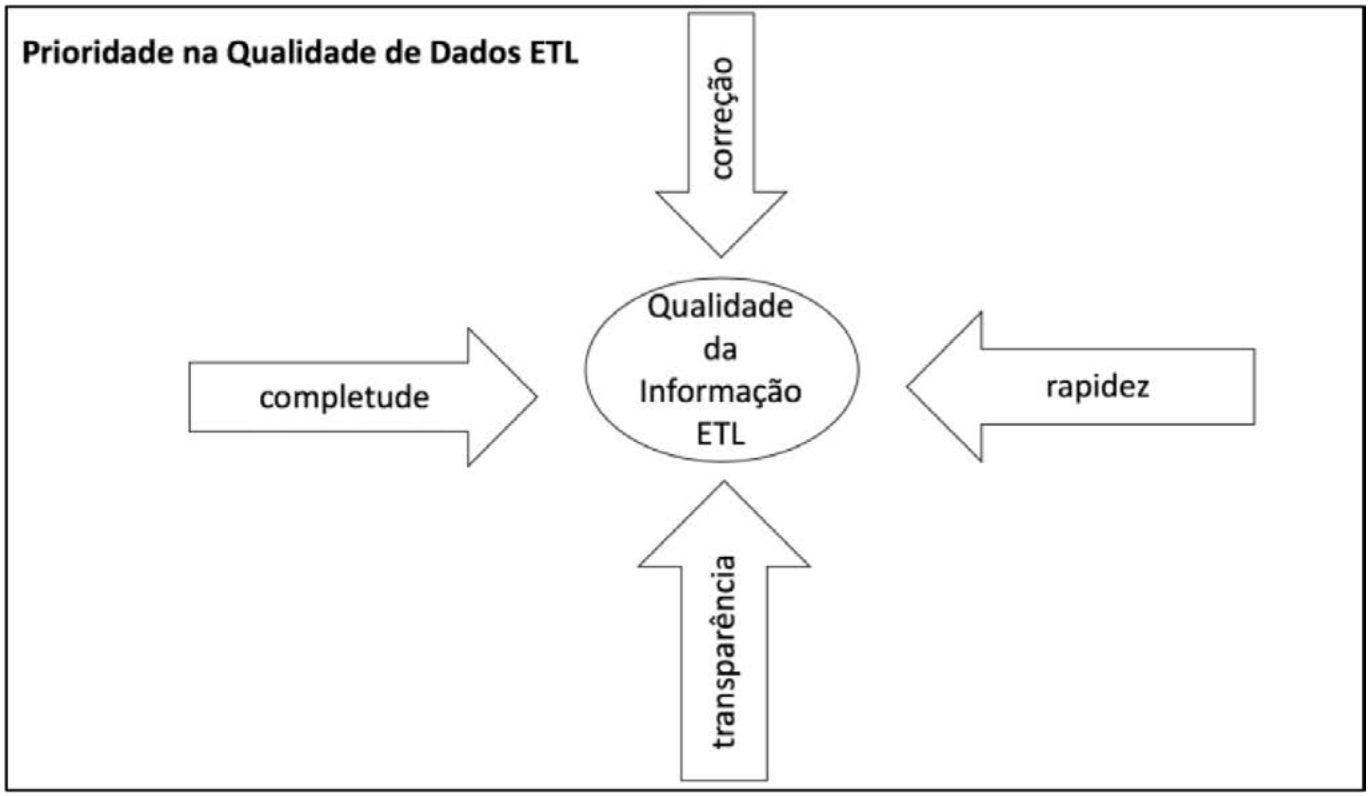
- Rapidez: deve ser processado um volume cada vez maior de dados em intervalos de tempo cada vez menores.

- Correção: corrigir problemas de qualidade de dados o mais próximo possível da fonte é, evidentemente, uma maneira estrategicamente defensável de melhorar os ativos de informação da organização, reduzindo assim altos custos e oportunidades perdidas devido à má qualidade dos dados. No entanto, a realidade é que muitas organizações ainda não estabeleceram ambientes formais de qualidade de dados ou líderes de qualidade da informação; nesses casos, a equipe de *data warehouse* pode ser a primeira a descobrir problemas de qualidade que estão se deteriorando há anos.

- Transparência: os relatórios e análises criadas usando o *data warehouse* devem expor defeitos e chamar a atenção para sistemas e práticas de negócios que prejudicam a qualidade dos dados da organização; essas revelações, em última análise, impulsionam a reengenharia de processos de negócios, na qual os sistemas de origem e os procedimentos de entrada de dados são aprimorados; realizar medidas heróicas para mascarar defeitos de qualidade na fonte pode ser uma situação em que o remédio pode ser pior que a doença.

A Figura 8 ilustra a prioridade na qualidade de dados ETL que tem como objetivo determinar dados completos, corretos, rápidos e transparentes.

Figura 8 – Prioridade na qualidade dos dados



Fonte: adaptado de Kimball e Caserta (2009, p. 119).

Segundo Kimball e Caserta (2009), um compromisso sério para melhorar a qualidade dos dados deve ser baseado em medições rigorosas, incluindo manutenção de registros precisos dos tipos de problemas de qualidade de dados. Algumas perguntas sobre a qualidade dos dados descobertos devem ser respondidas:

- A qualidade dos dados está melhorando ou piorando?
- Quais sistemas de origem geram mais ou menos problemas de qualidade de dados?
- Existem padrões ou tendências interessantes revelados no exame das questões de qualidade de dados ao longo do tempo?
- Existe alguma correlação observável entre os níveis de qualidade de dados e o desempenho da organização como um todo?

Segundo Kimball e Caserta (2009), a limpeza de dados deve ser iniciada após a extração dos dados. Deve ser realizada uma análise abrangente do perfil de dados e de suas fontes durante a fase inicial de planejamento e design.

Uma boa análise de perfil de dados assume a forma de um repositório de metadados específico, descrevendo:

- Definições de esquema;
- Objetos de negócios;
- Domínios;
- Fontes de dados;
- Definições de tabelas;
- Sinônimos;
- Regras de dados;
- Regras de valor.

Segundo Kimball e Caserta (2009), uma forma de atingir esse equilíbrio é realizar algumas perguntas sobre latência e qualidade dos dados em seu *data warehouse* a ser construído:

- Em que ponto a rigidez da entrega dos dados é definido?
- Quão importante é obter os dados verificadamente corretos?
- E se tivesse que determinar, por exemplo, um maior grau de confiança na qualidade dos dados e um atraso de um dia na publicação?

As pessoas que usam o *data warehouse* que publica diariamente pode, por exemplo, optar por negociar um dia inteiro de latência para obter confiança adicional na qualidade de dados, talvez por meio de testes expandidos de variação estatística ou padronização e correspondência de dados ou mesmo revisão/auditoria manual seletiva.

De acordo com Han e Kamber (2006), pode-se deparar com valores faltantes em uma base de dados:

- Por problemas no preenchimento e armazenamento dos dados;
- Em situações em que os dados não devem estar presentes: considere uma base que, ao armazenar dados de pacientes, um possível dado faltante pode ser o resultado de um "Exame B" que só é realizado caso o resultado de um "Exame A" seja positivo.

Segundo Han e Kamber (2006), algumas técnicas de preenchimento de dados podem ser aplicadas a valores faltantes:

- Descarte de toda a tupla: pode gerar a perda de muitas tuplas caso os dados faltantes estejam distribuídos por vários atributos; só é eficaz se a tupla tiver vários atributos faltantes.
- Preenchimento manual do valor faltante: demanda muito tempo e pode não ser factível em bases grandes.
- Uso de uma constante global para preencher o valor faltante: programas de mineração podem, erroneamente, identificar padrões considerados "interessantes" a partir dessa constante.
- Uso da média do atributo como valor a ser substituído: pode introduzir tendências indesejáveis nos dados (*bias*).
- Uso da média do atributo, calculada para todas as amostras de uma mesma classe, para substituir valores faltantes em amostras dessa classe: similar à estratégia anterior, mas direcionada para grupos específicos de dados (classes); também pode introduzir tendências indesejáveis.

- Uso do valor mais provável do atributo como valor a ser preenchido: exige o uso de técnicas adicionais para tal inferência, tais como regressão e árvores de decisão.

De acordo com Han e Kamber (2006), um ruído (*noise*) é uma variação ou erro aleatório observado em uma variável medida, podendo introduzir erros nos resultados. Existem algumas técnicas para "suavizar" os dados (remover o ruído), que acabam reduzindo o número de valores distintos existentes na base de dados para cada atributo ou discretizando os valores do atributo.

A compartimentalização (*binning*) é uma técnica de remoção de ruído que suaviza dados ordenados a partir dos dados em posições vizinhas. Distribui os dados ordenados em compartimentos (*bins*). Como são consultados valores vizinhos a cada dado, diz-se que é uma estratégia de suavização local. Existem diferentes técnicas de *binning* e todas começam com a etapa de particionamento.

Considere os dados ordenados para preço (em reais): 4, 8, 15, 21, 21, 24, 25, 28, 34. A partição em *bins*:

- *bin* 1: 4, 8, 15;
- *bin* 2: 21, 21, 24;
- *bin* 3: 25, 28, 34.

A primeira estratégia, proposta por Han e Kamber (2006), é a suavização pela média (mediana) dos compartimentos. Os valores são substituídos pela média (mediana) calculada para cada *bin*. Assim, a suavização ficaria:

- *bin* 1: 9, 9, 9;
- *bin* 2: 22, 22, 22;
- *bin* 3: 29, 29, 29.

A segunda estratégia é a suavização pelos valores de fronteira. Valores mínimo e máximo são identificados como valores de fronteira. Valores são substituídos pelo valor de fronteira mais próximo. Assim, a suavização ficaria:

- *bin* 1: 4, 4, 15;
- *bin* 2: 21, 21, 24;
- *bin* 3: 25, 25, 34.

ASSIMILE

Algumas tarefas executadas durante a limpeza (ou pré-preocessamento) dos dados são:

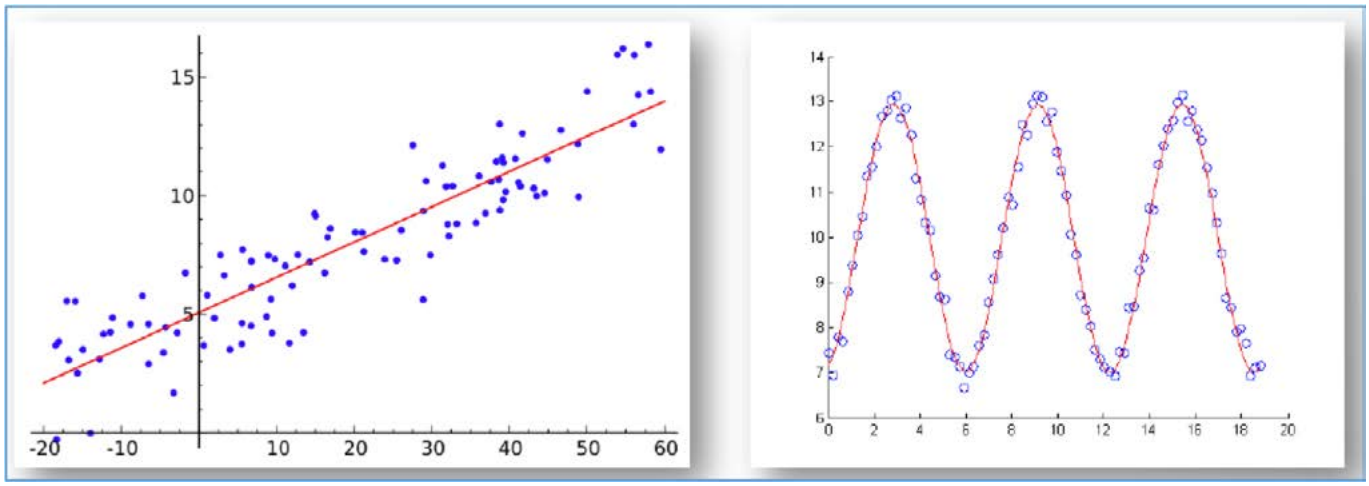
- Exclusão ou substituição de valores duplicados;
- Identificação, exclusão ou tratamento de valores nulos ou inconsistentes;
- Adequação na distribuição dos dados, a partir da identificação e do tratamento de valores atípicos.

Outra técnica para remoção de ruídos, segundo Han e Kamber (2006), a regressão, consiste em suavizar os dados substituindo-os pelo resultado de uma função que os aproxime. Essa função pode ser:

- Regressão linear: aproxima os dados por uma reta, plano ou hiperplano (conforme a dimensão dos dados).
- Regressão não linear: dados são aproximados por outras funções.

A Figura 9 apresenta a regressão linear e não linear, na qual os dados são aproximados por uma reta e por um hiperplano.

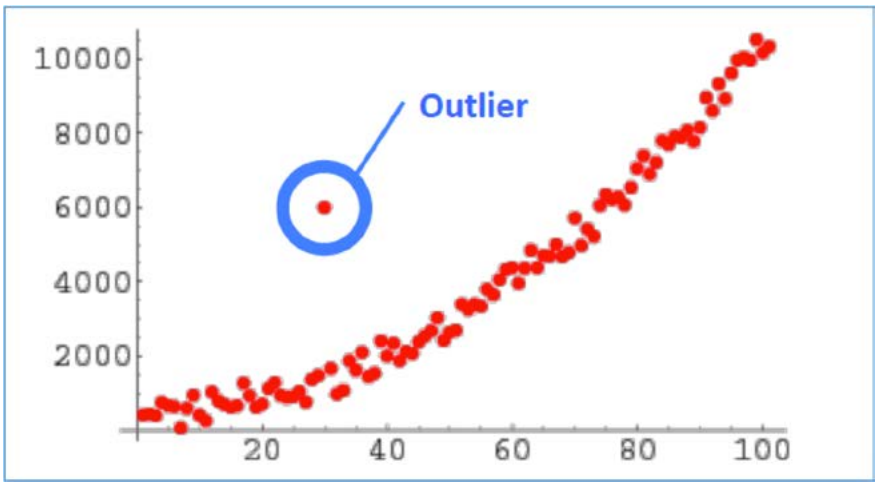
Figura 9 – Regressão linear e não linear



Fonte: adaptado de Han e Kamber (2006).

O agrupamento (*clustering*), segundo Han e Kamber (2006), é utilizado principalmente para eliminar *outliers*, que são valores "espúrios" que não seguem o comportamento geral ou o modelo dos dados. Geralmente são causados por erros na coleta dos dados. A Figura 10 ilustra um ponto (*outlier*) que não segue o comportamento geral ou o modelo dos dados.

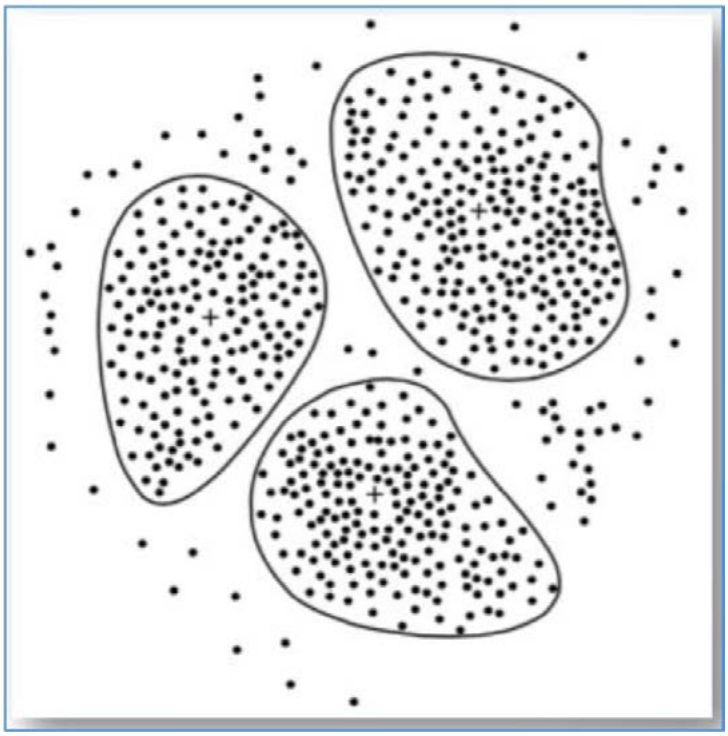
Figura 10 – Outlier



Fonte: adaptado de Han e Kamber (2006).

Na técnica de agrupamento (*clustering*), os dados são automaticamente divididos em grupos (clusters), e pontos que não pertencem a qualquer dos grupos são eliminados. A Figura 11 ilustra os dados divididos em grupos (clusters). Os pontos que não pertencem a qualquer dos grupos serão eliminados.

Figura 11 – Agrupamento (clustering)



Fonte: adaptado de Han e Kamber (2006).

Segundo Han e Kamber (2006), uma das etapas mais importantes na limpeza dos dados é detectar discrepâncias nos dados. Discrepâncias podem ter diferentes causas. Dentre elas:

- Formulários de entrada de dados mal projetados;
- Erros deliberados (usuários que não querem fornecer informações);
- Envelhecimento dos dados (mudança de endereço, telefone, etc);
- Problemas em sensores;
- Erros nos dispositivos de armazenamento.

De acordo com Han e Kamber (2006), para detectar discrepâncias, deve-se usar todo o conhecimento adicional que se tem sobre os dados (metadados), como domínio de cada atributo, tipo do dado e dependência entre os dados.

Deve-se procurar por erros de códigos, formatos e armazenamentos:

- Exemplo: valores 2015/12/25 e 25/12/2015 em campos de data;
- Valor inteiro que só pode ser representado em 32 bits em um campo de 16 bits;
- Verificar situações em que valores devem ser únicos ou explicitamente faltantes (*null values*).

Como suporte à detecção de discrepâncias, há ferramentas computacionais que realizam as seguintes atividades:

- Buscam e corrigem valores automaticamente, como verificadores de CEPs, corretores ortográficos, etc.;
- Analisam os dados e tentam estabelecer relações e regras, indicando as amostras que violam tais regras/relações:
  - São variantes de ferramentas de mineração de dados;
  - Exemplo: identificar correlações entre os atributos ou realizar *clustering* para identificar *outliers*.

PARA SABER MAIS

É importante lembrar que a limpeza dos dados também faz parte do controle de qualidade e, por isso, pode-se concluir que esse pré-processamento figura como uma etapa essencial para que os resultados de uma análise sejam confiáveis.

Negligenciar esse passo pode comprometer total ou parcialmente as etapas subsequentes da análise de dados. Portanto, recomenda-se que o processo seja conduzido com muita atenção e de forma criteriosa.

Dessa forma, você aprendeu as definições e os conceitos de limpeza de dados, bem como remover “ruídos”, dados irrelevantes e corrigir inconsistências nos dados, além de conhecer as técnicas de remoção de “ruídos” conhecidas como compartimentalização (*binning*), regressão e agrupamento (*clustering*).

TEORIA EM PRÁTICA

Uma empresa nacional de revenda de cosméticos está enfrentando alguns problemas financeiros. Dado o grande volume de produtos lançados pelo setor de cosméticos, a empresa está tendo dificuldades em acompanhar a demanda por produtos lançados, o que muitas vezes acaba por comprometer o resultado financeiro, por investir em produtos com baixa procura. Outro problema são os produtos que possuem prazo de validade curto, que acabam por vencer e não podem ser trocados pelos fornecedores. Hoje, a empresa conta com um sistema de vendas e controle de estoque, com banco de dados relacional e um processo de marketing pelas redes sociais, porém está tendo dificuldades para a tomada de decisões relacionadas ao que o seu público-alvo realmente consome e assim evitar gastos desnecessários. Após o processo de extração de dados do sistema de vendas e controle de dados e do marketing das redes social, constatou-se que os dados necessitavam passar por um processo de limpeza. Como podemos auxiliar a organização no processo de limpeza dos dados?

VERIFICAÇÃO DE LEITURA

1. Muitas vezes, os dados estão em grandes quantidades (GB ou mais), apresentando dificuldades de manipulação e nem todos os dados são necessários para a mineração. Considere a seguinte afirmação: “Os valores e descrições nos dados devem ter apenas um significado”. Assinale a alternativa que apresenta corretamente a que tipos de dados se refere a afirmação realizada anteriormente.

a. Dados não ambíguos.

b. Dados corretos.

c. Dados consistentes.

d. Dados completos.

e. Dados ambíguos.

2. Considere a seguinte afirmação: “Os valores e descrições nos dados descrevem seus objetos de verdade e devem estar corretos”. Assinale a alternativa que apresenta, corretamente, a que tipos de dados se refere a afirmação realizada anteriormente.

a. Dados não ambíguos.

b. Dados corretos.

c. Dados consistentes.

d. Dados completos.

e. Dados ambíguos.

3. Considere a seguinte afirmação: “Garantir que os valores e descrições individuais nos dados são definidos (não nulos) para cada instância”. Assinale a alternativa que apresenta, corretamente, a que tipos de dados se refere a afirmação realizada anteriormente.

a. Dados não ambíguos.

b. Dados corretos.

c. Dados consistentes.

d. Dados completos.

e. Dados ambíguos.

Referências bibliográficas

HAN, J.; KAMBER, M. **Data Mining:** concepts and techniques. Waltham: Elsevier, 2006.

KIMBALL, R.; CASERTA, J. **The Data Warehouse ETL Toolkit:** practical techniques for extracting, cleaning, conforming, and data delivering data. Indianopolis: Wiley Publishing, 2009.

Gabarito

Questão 1 – Resposta A

Os dados devem ser não ambíguos, apresentando os valores e descrições dos dados com apenas um significado.

Questão 2 – Resposta B

Os dados devem ser corretos, com valores e descrições nos dados descrevendo seus objetos de verdade e devem estar corretos.

Questão 3 – Resposta D

Os dados devem ser completos, garantindo que os valores e descrições individuais nos dados são definidos (não nulos) para cada instância.

Conformação de dados

Autor: Thiago Salhab Alves

Objetivos

- Compreender as definições e os conceitos de conformação de dados.
- Aprender a tratar problemas como redundância e conflito de valores.
- Aprender as técnicas de consolidação de dados.

1. Conformação de dados

Você já parou para pensar como as organizações nacionais e multinacionais realizam o processo de conformação dos dados após a extração e limpeza dos dados para compor um *data warehouse*? Diversas são as fontes de dados que apresentam dados heterogêneos e que necessitam ser combinados para compor uma base de dados unificada. Como adotar um processo de conformação para essas diversas fontes de dados de origem? Dessa forma, faz-se necessário conhecer o processo de conformação de dados.

Esta leitura irá apresentar as definições e os conceitos de conformação de dados. Você aprenderá a tratar problemas como redundância e conflito de valores, além de técnicas de consolidação de dados.
Uma boa aula e bom trabalho.

1.1 Processo de conformação de dados

De acordo com Kimbal e Caserta (2009), a conformação ou integração de dados diz respeito à criação de dimensões e instâncias de fatos configuradas, combinando as melhores informações de várias fontes de dados em uma visão mais abrangente. Para fazer isso, os dados recebidos precisam ser estruturalmente idênticos, filtrados de registros inválidos, padronizados em termos de conteúdo e não duplicados.

De acordo com Han e Kamber (2006), a conformação de dados é a combinação de dados de diferentes fontes em uma base de dados única e coerente.

Problemas:

- Identificação de entidades: como garantir que um atributo presente em duas fontes tenha o mesmo significado?
    - Exemplo: costumer_id.

- Quais valores os campos podem assumir?
    - Exemplo: campo “sexo” pode ter valores “H/M” ou “M/F”.
- Redundância:
    - Dados duplicados;
    - Vários atributos podem vir a ser obtidos a partir de outro atributo ou conjunto de atributos. Seu armazenamento aumenta o espaço necessário para armazenamento e aumenta a chance de se terem dados inconsistentes;
    - Pode-se tentar identificar redundância a partir de análises de correlações.
- Conflito de valores:
    - Um mesmo atributo pode ter valores distintos em fontes diferentes;
    - Codificação diferente;
    - Unidades e contextos diferentes:
        - Exemplo: peso em kg e em g;
        - Exemplo: diárias de hotel com e sem café da manhã.

Segundo Han e Kamber (2006), a transformação de dados consiste em transformar ou consolidar os dados em um formato mais adequado para o *data warehouse*. Vários tipos de transformação são possíveis:

- Suavização: visa eliminar ruídos.
- Agregação: operações de resumo ou agregação são realizadas.
    - Exemplo: dados horários de chuva (em mm) são resumidos em um único atributo correspondente ao total acumulado em um dia.
- Generalização: consiste em substituir dados de “baixo nível” por dados de “alto nível”.

- Exemplo: idade passa a ser “jovem”, “adulto” ou “idoso”.

- Normalização: atributos são escalados para um novo intervalo mais adequado a ser usado.
  - Exemplo: [0;1] ou [-1;1].
  - Exemplo de problemas que a não normalização dos dados pode causar – cálculo da distância euclidiana:
    - Quatro amostras: w = [0,10; 1375,00], x = [0,00; 1000,00], y = [0,20; 1375,00] e z = [0,10; 2750,00];
    - $d(w, x) = [(0,10 - 0,00)^2 + (1375,00 – 1000,00)^2]^{0,5} = 375,0000133$;
    - $d(y, x) = [(0,20 – 0,00)^2 + (1375,00 – 1000,00)^2]^{0,5} = 375,0000533$;
    - $d(z, x) = [(0,10 – 0,00)^2 + (2750,00 – 1000,00)^2]^{0,5} = 1750,000003$;
  - Normalização min-max: normaliza os dados que originalmente pertenciam ao intervalo $[min_A; max_A]$ para o intervalo $[new\_min_A; new\_max_A]$;
  - Normalização: dados passam a ter média 0,0 e desvio padrão 1,0.

- Construção de atributos: novos atributos são construídos e adicionados ao conjunto de dados, para auxiliar o *data warehouse*.
  - Exemplo: criar um atributo volume, a partir dos atributos base, volume e profundidade.
  - A construção de atributos pode ajudar a estratégia de mineração de dados a evidenciar relações entre os atributos, que podem ser úteis no processo de descoberta de conhecimento.

De acordo com Han e Kamber (2006), a redução dos dados é uma técnica que busca obter uma representação significativamente menor dos dados (em volume), mas que mantenha a integridade dos dados originais.

Algumas estratégias de redução de dados:

- Seleção de atributos: atributos irrelevantes, pouco relevantes ou redundantes são removidos da base de dados.
- Redução de dimensão: técnicas de codificação são usadas para reduzir a dimensão (número de atributos) dos dados.
    - Envolve a transformação dos dados.
- Redução de número: dados são substituídos ou estimados por representações alternativas e menores.
    - Modelos paramétricos: são armazenados apenas os parâmetros do modelo.
    - Modelos não paramétricos: tais como *clustering*, amostragem ou histogramas – são armazenados os resultados desses modelos.
- Discretização de atributos:
    - É usada para reduzir o número de valores para um dado atributo contínuo;
    - Domínio do atributo é dividido em intervalos;
    - A cada intervalo é associado um rótulo;
    - Após a discretização, podem-se substituir os dados por categorias mais genéricas, que facilitam a interpretação dos dados.
        - Exemplo: atributo numérico idade substituído por “jovem”, “adulto” e “idoso”.

De acordo com Kimball e Caserta (2009), quando se conformam dados, podem-se converter os gêneros de (M, F), (H, M) e (Homem, Mulher) de três diferentes provedores de dados em um atributo de dimensão de gênero padrão (Masculino, Feminino).

A conformidade de atributos descritivos em várias fontes de dados e vários *data marts* que participam de um *data warehouse* distribuído é uma das principais etapas de desenvolvimento para o arquiteto do *data warehouse* e a equipe ETL. As preocupações imediatas da equipe de ETL estão em capturar todas as entradas sobrepostas e conflitantes e suportando as necessidades do gerenciador de dimensão e do provedor de tabela de fatos (principal tabela do *data warehouse*, armazenando as métricas, que são os fatos propriamemente ditos e as chaves estrangeiras, que servem para ligar os dados das dimensões com o fato) (KIMBALL; CASERTA, 2009).

Independentemente da arquitetura de hardware, cada *data warehouse* é distribuído em um certo sentido, porque tipos separados de medidas devem sempre existir em tabelas de fatos separados. A mesma afirmação é verdadeira em um ambiente modelado em ER (Entidade Relacionamento). Portanto, para que um aplicativo de usuário final combine dados de tabelas de fatos separados, devem-se implementar interfaces consistentes para essas tabelas de fatos, para que os dados possam ser combinados. Essas interfaces consistentes apresentam dimensões conformadas e fatos conformados (KIMBALL; CASERTA, 2009).

Exemplos de dimensões frequentemente conformadas incluem cliente, produto, localização, oferta (promoção) e calendário (tempo). A principal responsabilidade da equipe central de design de *data warehouse* é estabelecer, publicar, manter e impor dimensões conformadas. Uma dimensão de cliente conformada é uma tabela mestre de clientes com uma chave de cliente substituta (que é o campo de *primary key* da dimensão) limpa e muitos atributos bem mantidos que descrevem cada cliente.

Os campos de endereço na dimensão do cliente, por exemplo, devem constituir o melhor endereço para correspondência conhecido para cada cliente... É de responsabilidade da equipe central de *data warehouse* criar a dimensão do cliente conformada e fornecê-la como um recurso para o restante da empresa.

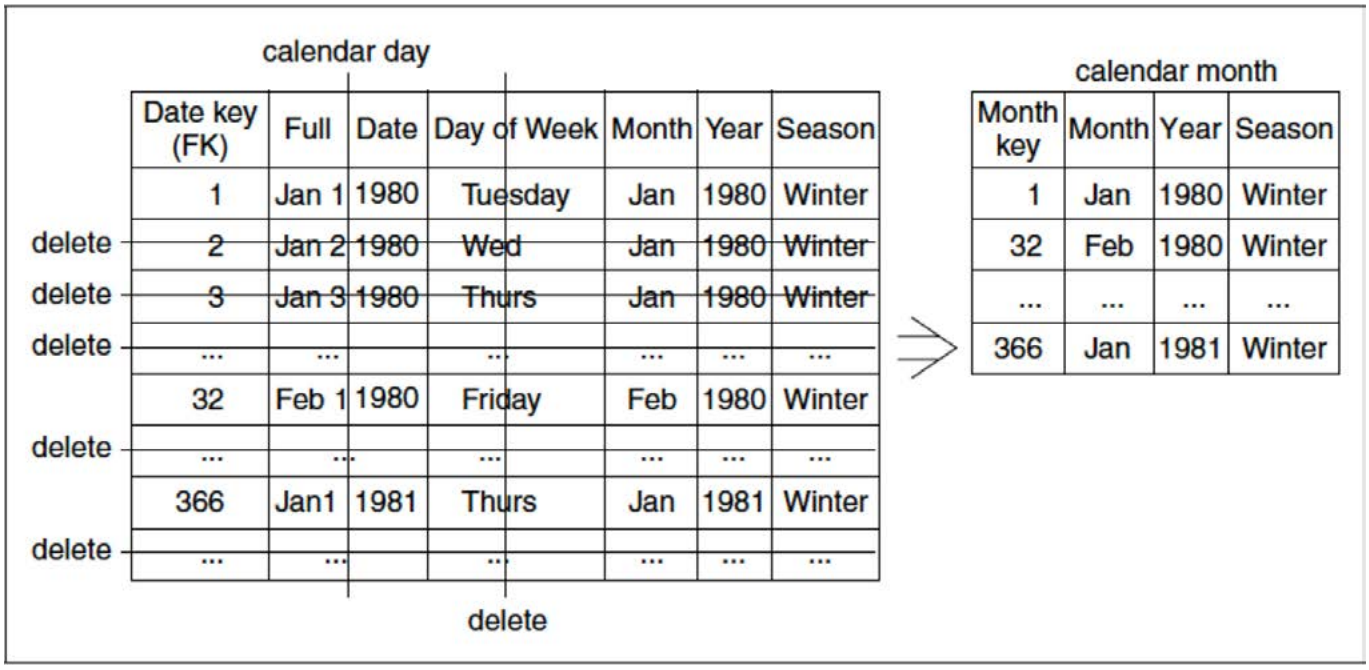
Segundo Kimball e Caserta (2009), as dimensões conformadas são extremamente importantes para o *data warehouse*. Sem uma adesão restrita às dimensões conformadas, o *data warehouse* não pode funcionar como um todo integrado. Se uma dimensão como cliente ou produto for usada de maneira não conforme, as tabelas de fatos separadas simplesmente não podem ser usadas juntas ou, na tentativa de usá-las juntas, produzirão resultados errados.

De acordo com Kimball e Caserta (2009), se a equipe central de *data warehouse* conseguir definir e fornecer um conjunto de dimensões conformes para a empresa, é extremamente importante que os proprietários de tabelas de fatos separadas usem essas dimensões. O compromisso de usar as dimensões conformes é muito mais que uma decisão técnica, sendo uma decisão política comercial que é fundamental para tornar funcional o armazenamento de dados da empresa. O uso das dimensões conformadas deve ser suportado nos mais altos níveis executivos.

É possível criar um subconjunto de uma tabela de dimensões conformada para determinadas tabelas de fatos se tiver conhecimento que o domínio da tabela de fatos associada contém apenas esse subconjunto. Por exemplo, a tabela de produtos mestre pode ser restrita apenas aos produtos fabricados em um determinado local, se o *data mart* em questão se referir apenas a esse local. Pode-se chamar de um subconjunto de dados simples, pois a tabela de dimensões reduzidas preserva todos os atributos da dimensão original (KIMBALL; CASERTA, 2009).

Na Figura 12 um subconjunto de linhas e colunas da tabela original foi removida, restringindo a tabela de dimensões de datas de dias para meses, mantendo, assim, apenas o registro que descreve o primeiro dia de cada mês.

Figura 12 – Construindo uma tabela conformada de calendário



Fonte: KIMBALL; CASERTA, 2009, p. 151.

De acordo com Kimball e Caserta (2009), identificar as definições de fatos padrão é feito ao mesmo tempo que a identificação das dimensões conformadas. Estabelecer dimensões conformadas é um processo colaborativo em que as partes interessadas para cada tabela de fatos concordam em usar as dimensões conformadas.

Durante reuniões de conformidade, as partes interessadas também precisam identificar fatos semelhantes presentes em cada uma das tabelas de fatos. Por exemplo, várias tabelas de fatos podem relatar receita. Se os aplicativos do usuário final esperam adicionar ou comparar essas medidas de receita de tabelas de fatos separadas, as regras de negócios que definem essas medidas de receita devem ser as mesmas.

Segundo Kimball e Caserta (2009), os fatos conformados podem ser comparados diretamente e podem participar de expressões matemáticas, como somas ou proporções. Se as partes interessadas das tabelas de fatos puderem chegar a um acordo, as etapas de preparação

de dados para algumas ou todas as tabelas de fatos podem envolver transformações dos fatos para atender à definição comum.

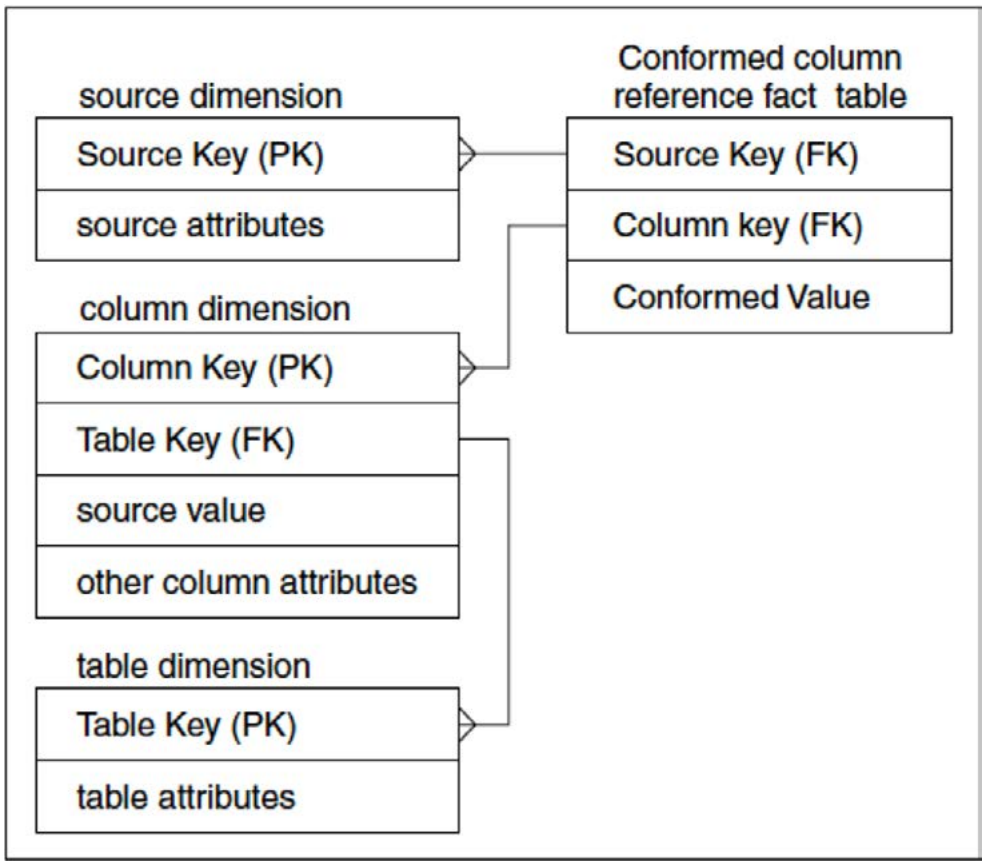
Para implementar dimensões e fatos conformados, o subsistema em conformidade precisa de metadados de referência que capturam os relacionamentos entre valores explicitamente válidos de sistemas de origem para valores de atributos de dimensão conformados e valores de fatos conformados. Muitas ferramentas ETL fornecem suporte a esses tipos de mapeamentos de domínio, com atributos de metadados pré-construídos ou permitindo que a equipe ETL use atributos de metadados extensíveis para os objetos da tabela de origem.

Na Figura 13, as entidades da tabela e colunas capturam metadados sobre cada tabela e suas colunas associadas, respectivamente. A tabela de fatos registra os valores conforme oficialmente definido em cada sistema de origem. As entidades dos sistemas de origem gerais são capturadas na tabela do sistema de origem.

A dimensão da coluna contém o valor de origem mapeado no valor oficial em conformidade. Como exemplo, considere se Masculino e Feminino fossem valores conformes para o gênero, a tabela de fatos associaria M para Masculino e F a Feminino do sistema de origem A; e M para Masculino mas M (Mulher) com Feminino no sistema B; e H (Homem) com Masculino e M (Mulher) com Feminino. Colunas em registros que contêm valores inválidos, ou seja, valores que não estão no conjunto de valores válidos explícitos na tabela de dimensão da coluna, devem ser substituídas por um valor predefinido como “desconhecido” na tabela de referência de valor padronizado e a substituição deve ser anotada na tabela de fatos do evento de erro.

A Figura 13 apresenta um exemplo de tabelas de metadados que suportam dados em conformidade.

Figura 13 – Esquema de coluna em conformidade



Fonte: KIMBALL; CASERTA, 2009, p. 155.

ASSIMILE

É importante que dados falsos ou inválidos, que não podem ser padronizados, sejam removidos da visibilidade de processos ETL. Em outros casos, nenhuma correspondência definitiva é encontrada, e a única pista disponível para eliminar a duplicação é a similaridade de várias colunas quase iguais.

Segundo Kimball e Caserta (2009), as ferramentas de integração de dados conseguem lidar com a padronização de dados. O software deve comparar o conjunto de registros no fluxo de dados ao universo de registros de dimensão conformados e retornar:

- Pontuação numérica que quantifica a probabilidade de uma correspondência;
- Conjunto de chaves de correspondência que vinculam os registros de entrada a instâncias de dimensão conformada e/ou dentro do universo de registro padronizado sozinho.

Assim, a execução de registro de entrada por meio dos processos de correspondência pode ser uma correspondência para zero ou um registro de dimensão conformado e zero, um ou mais outros registros de entrada na fila de processo em lote. Em ambos os casos, a tarefa do software de correspondência é associar chaves que detalham essas relações de correspondências derivadas a esses registros de entrada (KIMBALL; CASERTA, 2009).

Kimball e Caserta (2009) ainda descrevem que muitas ferramentas de correspondência de dados também incluem uma pontuação de correspondência ou uma métrica de confiança correspondente, que descreve a probabilidade de correspondência obtida. Frequentemente, essas pontuações de correspondência são derivadas pela criação de várias abordagens correspondentes, pontuando probabilidades de correspondência de cada passagem e, em seguida, destilando os resultados em um conjunto recomendado de chaves de correspondência e uma pontuação ponderada geral.

As organizações com necessidades de recursos para não duplicação de dados podem optar por manter uma biblioteca persistente de dados correspondidos anteriormente, cada um ainda associado a um único provedor de dados, e usar essa biblioteca consolidada para melhorar os resultados correspondentes. Dessa forma, o mecanismo de correspondência pode aplicar sua correspondência não apenas aos registros de dimensão conformados, mas também ao conjunto completo de registros de dimensão correspondidos anteriormente que possui de todos os sistemas de origem (KIMBALL; CASERTA, 2009).

Essa abordagem pode resultar em correspondências melhores, porque o universo de candidatos a correspondência é mais rico e é muito mais resiliente para lidar com as mudanças de regras correspondentes, que agora podem ser satisfeitas. Essa abordagem dificulta o processamento de correspondências porque as correspondências podem ocorrer dentro e entre os universos de dados de origem.

A última etapa do processo de conformação de dados é chamada de Survivorship (sobrevivência), que se refere ao processo de destilação de um conjunto de registros correspondentes (não duplicados) em uma imagem unificada que combina os valores de coluna da mais alta qualidade em cada um dos registros correspondentes, para criar registros de dimensão conformados. Isso envolve o estabelecimento de regras de negócios que definem uma hierarquia para seleções de valores de coluna de todas as origens possíveis e a captura do mapeamento de origem-alvo a ser aplicado ao gravar registros sobreviventes (conformados) (KIMBALL; CASERTA, 2009).

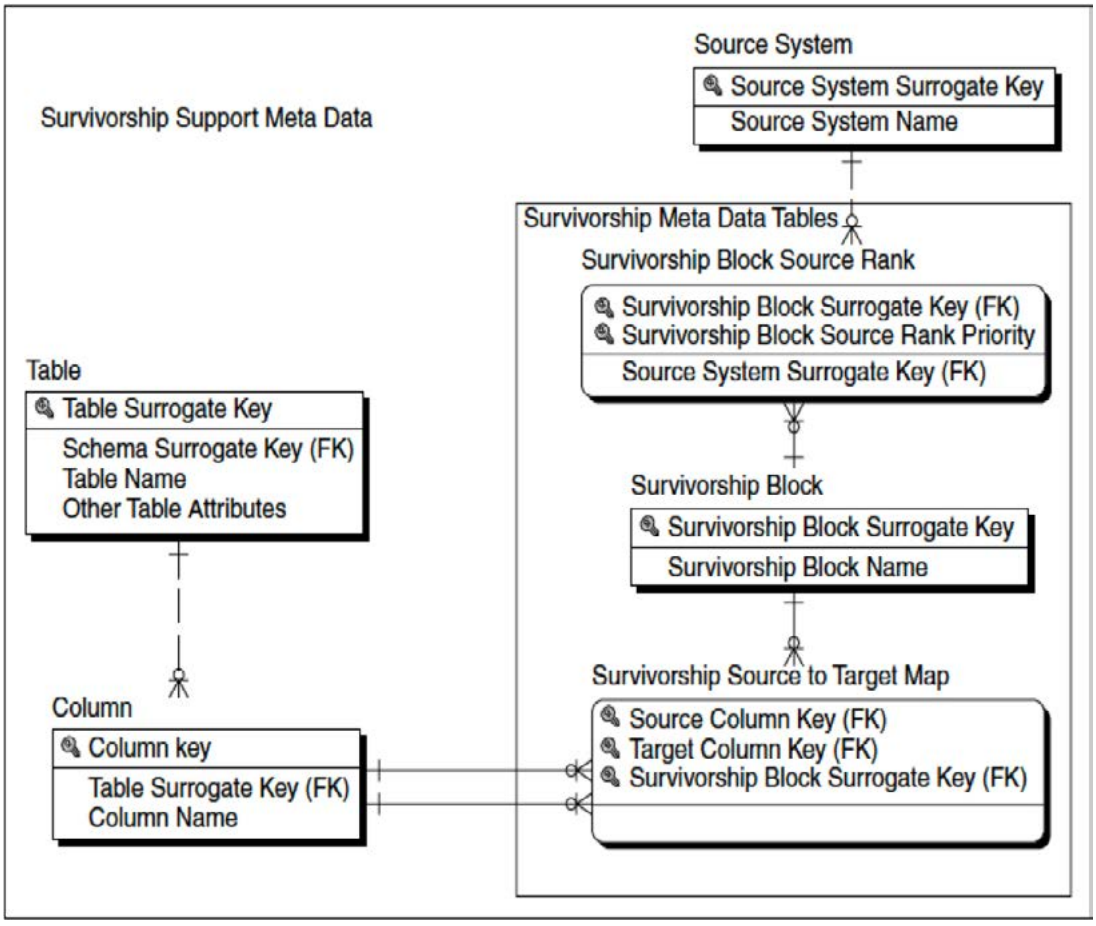
Além disso, de acordo com Kimball e Caserta (2009), a sobrevivência deve ser capaz de destilar combinações de colunas juntas em vez de individualmente. Isso é necessário para situações em que a combinação de colunas individualmente sobrevividas pode resultar em uma combinação “absurda”, como tentar combinar as linhas de endereço 1, 2 e 3 de três sistemas de origem diferentes e terminar com um endereço destilado que seja menos confiável do que todos os três. A Figura 14 apresenta os seguintes elementos:

- Fonte de sobrevivência para o mapa de destino: a tabela origem de sobrevivência para o mapa de destino captura mapeamentos de integração de dados entre colunas de origem (dados de entrada que foram limpos, mas que não estão conformados) e colunas de destino (colunas de tabela de dimensões conformadas). Para flexibilidade e simplicidade, ele permite que qualquer combinação de colunas seja usada como origem em qualquer combinação de alvos, sobrecarregando o arquiteto de ETL (em vez da integridade referencial que poderia ter sido incluída em uma estrutura mais complexa) para preenchê-lo.

- Tabela de bloco de sobrevivência: grupos que mapearam as colunas de origem e de destino em blocos que devem ser mantidos juntos. Os blocos de sobrevivência podem ter apenas uma origem e um destino, portanto, ao forçar toda a sobrevivência a ser executado por bloco, pode-se simplificar o modelo de metadados e o processamento de sobrevivência. Esta tabela inclui uma classificação que permite que a prioridade dos blocos de campos do sistema de origem seja determinada com SQL dinâmico, que procura valores não nulos em cada bloco ordenado pela prioridade de classificação de origem do bloco de sobrevivência e constrói uma instrução INSERT ou UPDATE apropriada, dependendo se a chave de correspondência já existe como uma chave substituta de registro conformado (UPDATE) ou não (INSERT).

A Figura 14 apresenta os requisitos mais comuns de sobrevivência suportados.

Figura 14 – Requisitos mais comuns de sobrevivência suportados



Fonte: KIMBALL; CASERTA, 2009, p. 159.

PARA SABER MAIS

Nos casos em que o processo de não duplicação combina com êxito entidades de origem separadas em uma única entidade, se as entidades de origem tiverem chaves primárias separadas no sistema de origem, uma tabela dessas chaves primárias obsoletas deve ser mantida para acelerar a não duplicação subsequente, usando dados desse sistema de origem.

Dessa forma, você aprendeu as definições e os conceitos de conformação de dados. Você aprendeu a tratar problemas como redundância e conflito de valores e técnicas para consolidação de dados.

TEORIA EM PRÁTICA

Uma empresa nacional de revenda de cosméticos está enfrentando alguns problemas financeiros. Dado o grande volume de produtos lançados pelo setor de cosméticos, a empresa está tendo dificuldades em acompanhar a demanda por produtos lançados, o que muitas vezes acaba por comprometer o resultado financeiro, por investir em produtos com baixa procura. Outro problema são os produtos que possuem prazo de validade curto, que acabam por vencer e não podem ser trocados pelos fornecedores. Hoje, a empresa conta com um sistema de vendas e controle de estoque, com banco de dados relacional e um processo de marketing pelas redes sociais, porém está tendo dificuldades para a tomada de decisões relacionadas ao que o seu público-alvo realmente consome e, assim, evitar gastos

desnecessários. Após o processo de extração e limpeza dos dados do sistema de vendas e controle de dados e do marketing das redes sociais, constatou-se que os dados necessitavam passar por um processo de consolidação.

Como podemos auxiliar a organização no processo de consolidação dos dados?

VERIFICAÇÃO DE LEITURA

1. Considere a seguinte afirmação: “Diz respeito à criação de dimensões e instâncias de fatos configuradas, combinando as melhores informações de várias fontes de dados em uma visão mais abrangente. Para fazer isso, os dados recebidos precisam ser estruturalmente idênticos, filtrados de registros inválidos, padronizados em termos de conteúdo e não duplicados”. Assinale a alternativa que apresenta, corretamente, a que processo a afirmação se refere.

   a. Processo de extração de dados.

   b. Processo de transformação de dados.

   c. Processo de carregamento de dados.

   d. Processo de limpeza de dados.

   e. Processo de conformação de dados.

2. De acordo com Han e Kamber (2006), a conformação de dados é a combinação de dados de diferentes fontes em uma base de dados única e coerente. Um dos problemas da conformação de dados é que “um

mesmo atributo pode ter valores distintos em fontes diferentes". Assinale a alternativa que apresente, corretamente, o problema apresentado.

a. Redundância.

b. Conflito de valores.

c. Identificação de entidades.

d. Valores que os campos podem assumir.

e. Unidades diferentes.

3. Segundo Han e Kamber (2006), a transformação de dados consiste em transformar ou consolidar os dados em um formato mais adequado para o *data warehouse*. Um dos tipos de transformação consiste em substituir dados de "baixo nível" por dados de "alto nível". Assinale a alternativa que apresente, corretamente, o tipo de transformação a qual o texto se refere:

a. Suavização.

b. Agregação.

c. Generalização.

d. Normalização.

e. Construção de atributos.

Referências bibliográficas

HAN, J.; KAMBER, M. **Data Mining:** concepts and techniques. Waltham: Elsevier, 2006.

KIMBALL, R.; CASERTA, J. **The Data Warehouse ETL Toolkit:** practical techniques for extracting, cleaning, conforming, and data delivering data. Indianopolis: Wiley Publishing, 2009.

Gabarito

Questão 1 – Resposta E

O processo de conformação de dados se há criação de dimensões e instâncias de fatos configuradas, combinando as melhores informações de várias fontes de dados em uma visão mais abrangente.

Questão 2 – Resposta B

O conflito de valores é aquele em que um mesmo atributo pode ter valores distintos em fontes diferentes.

Questão 3 – Resposta C

A generalização consiste em substituir dados de “baixo nível” por dados de “alto nível”.

Entrega de dados

Autor: Thiago Salhab Alves

Objetivos

- Compreender o processo de entrega de dados.
- Aprender sobre dimensões planas e dimensões flocos de neve.
- Aprender sobre dimensões de data e hora e dimensões grandes.

1. Entrega de dados

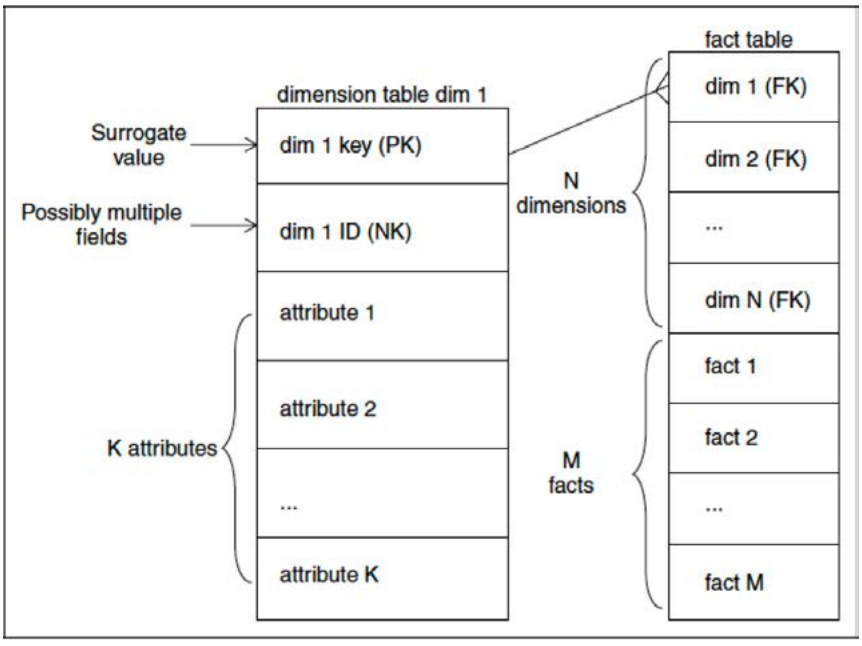
Prezado aluno, você já parou para pensar em como as organizações, nacionais e multinacionais, realizam o processo de entrega dos dados após a extração, limpeza e conformação dos dados para compor um *data warehouse*? Diversas são as fontes de dados que apresentam dados heterogêneos e que necessitam ser combinados para compor uma base de dados unificada. Como adotar um processo de entrega dos dados? Dessa forma, faz-se necessário conhecer o processo de entrega de dados.

Esta leitura irá apresentar os conceitos do processo de entrega de dados. Você aprenderá sobre dimensões planas e dimensões flocos de neve, dimensões de data e hora e dimensões grandes. Uma boa aula e bom trabalho.

1.1 Processo de entrega de dados

De acordo com Kimbal e Caserta (2009), a etapa final do processo de ETL é a entrega de dados, em que são preparadas as estruturas de tabelas dimensionais de forma mais restrita. Todas as dimensões devem ser fisicamente contruídas para obter o conjunto mínimo de componentes, conforme apresentado pela Figura 15.

Figura 15 – A estrutura básica de uma dimensão



Fonte: KIMBALL; CASERTA, 2009, p. 162.

Segundo Kimball e Caserta (2009), a chave primária fica armazenada em um único campo contendo um valor inteiro único chamado valor substituto. O processo de ETL do *data warehouse* deve sempre criar e inserir as chaves substitutas. O *data warehouse* possui essas chaves e nunca permite que outra entidade as atribua. A chave primária de uma dimensão é usada para unir a tabela de fatos, que deve preservar a integridade referencial. A chave de dimensão principal é unida a uma chave estrangeira correspondente na tabela de fatos.

Obtém-se um melhor desempenho na maioria dos bancos de dados relacionais quando todas as junções entre tabelas de dimensão e tabelas de fatos são baseadas nessas junções de número inteiro de campo único. De acordo com Kimbal e Caserta (2009), todas as tabelas de dimensões devem possuir um ou mais campos que compõem a chave natural da dimensão. Na Figura 15, o ID é designado campo de chave natural com NK. A chave natural é baseada em um ou mais campos significativos extraídos do sistema de origem.

Considere, por exemplo, uma simples dimensão de empregado que provavelmente teria o campo EMP_ID, que provavelmente é o número do empregado. EMP_ID seria a chave natural dessa dimensão de empregado. Mesmo com a chave EMP_ID, uma chave substituta deve ser atribuída ao *data warehouse*.

O componente final de todas as dimensões, de acordo com Kimball e Caserta (2009), além da chave primária e da chave natural, é o conjunto de atributos descritivos. Os atributos descritivos são predominantemente textuais, mas os atributos descritivos numéricos são legítimos. O arquiteto do *data warehouse* provavelmente especificará um grande número de atributos descritivos para dimensões, como empregado, cliente e produto.

Segundo Kimball e Caserta (2009), criar chaves substitutas por meio do SGBD é provavelmente a técnica mais comum usada atualmente. No entanto, essa tendência está mudando. No passado, era prática comum ter chaves substitutas criadas e inseridas por acionadores de banco de dados (*triggers*). Posteriormente, foi determinado que os gatilhos (acionadores) causam gargalos graves no processo de ETL e devem ser eliminados de quaisquer novos processos que estão sendo criados. Mesmo que ainda seja aceitável que os números inteiros de uma chave substituta sejam mantidos pelo DBMS, esses números inteiros devem ser criados diretamente pelo processo ETL. Ter o processo ETL pede ao SGBD um novo valor para a sequência do banco de dados, produzindo uma melhoria significativa no desempenho ETL sobre o uso de acionadores de banco de dados.

Além disso, o uso do banco de dados para gerar chaves substitutas quase garante que as chaves estarão fora de sincronia entre os diferentes ambientes de desenvolvimento, teste e produção do *data warehouse*. À medida que cada ambiente é carregado em intervalos diferentes, seu respectivo banco de dados pode gerar diferentes valores-chave substitutos para os mesmos registros de dimensão de entrada. Para maior eficiência, considere ter uma ferramenta ETL ou um aplicativo de terceiros para gerar e manter suas chaves substitutas. Certifique-se de que a geração e a manutenção eficiente de chaves substitutas estejam em seus critérios de sucesso de prova de conceito de ETL.

De acordo com Kimball e Caserta (2009), uma solução tentadora vista repetidamente durante as revisões de design é concatenar a chave natural do sistema de origem e um carimbo de data que reflete quando o registro foi criado no sistema de origem ou inserido no *data warehouse*. Dar à chave substituta a hora exata de sua criação pode ser útil em algumas situações, mas não é uma alternativa aceitável a uma chave substituta baseada em inteiro verdadeiro. Chaves inteligentes falham como uma chave substituta aceitável pelos seguintes motivos:

- Por definição: chaves substitutas devem ser sem sentido. Ao aplicar inteligência à chave substituta, sua responsabilidade é ampliada, fazendo com que elas precisem ser mantidas. O que acontece se uma chave primária no sistema de origem for alterada ou for corrigida de alguma forma? A chave inteligente concatenada precisaria ser atualizada, assim como todos os registros associados nas tabelas de fatos em todo o *data warehouse*.

- Atuação: concatenar a chave do sistema de origem com um carimbo de data degrada o desempenho da consulta. Como a equipe do *data warehouse* não tem controle sobre o conteúdo das chaves do sistema de origem e deve ser capaz de manipular qualquer tipo de dados, esse fato força a usar os tipos de dados CHAR ou VARCHAR para acomodar chaves alfa, numéricas ou alfanuméricas provenientes dos sistemas de origem. Além disso, ao anexar o carimbo de data à chave, potencialmente 16 caracteres ou mais, o campo pode se tornar pesado. O que é pior, essa chave precisará ser propagada em grandes tabelas de fatos em todo o *warehouse*. O espaço para armazenar os dados e índices seria excessivo, fazendo com que o desempenho de consulta do usuário final e do ETL diminuísse. Além disso, unir essas grandes colunas concatenadas VARCHAR durante o tempo de consulta será lento quando comparado à mesma junção usando colunas INTEGER.

- Incompatibilidade de tipos de dados: os modeladores de dados veteranos de *data warehouse* saberão construir as chaves substitutas do modelo dimensional com o tipo de dados NUMBER ou INTEGER.

- Dependência do sistema de origem: o uso da abordagem de chave inteligente depende do sistema de origem que revela exatamente quando um atributo em uma dimensão é alterado. Em muitos casos, essa informação simplesmente não está disponível. Sem a manutenção confiável de algum tipo de coluna de auditoria, a obtenção do registro de data e hora exato de uma alteração pode ser impossível.

- Fontes heterogêneas: a concatenação da chave natural e do carimbo de data suporta apenas um ambiente homogêneo. Em praticamente todos os *data warehouses* corporativos, as dimensões comuns são originadas por muitos sistemas de origem diferentes. Esses sistemas de origem têm seu próprio propósito e podem identificar de maneira diferente os mesmos valores de uma dimensão. A abordagem da chave natural concatenada e do registro de data é insuficiente com a introdução de um segundo sistema de origem. As chaves naturais de cada sistema devem ser armazenadas igualmente, em colunas não chave dedicadas na dimensão. Imagine tentar concatenar cada chave natural e seus respectivos *timestamps* (marcação de tempo) – um pesadelo de manutenção.

ASSIMILE

A característica atraente de usar essa estratégia de chave inteligente proibida é sua simplicidade no tempo de desenvolvimento de ETL. Ao construir o primeiro *data mart*, quando é bastante simples implementar uma chave inteligente anexando o SYSDATE (funções para manipular datas) à chave natural após a inserção. Evite a tentação desse atalho proibido.

De acordo com Kimball e Caserta (2009), algumas dimensões são criadas inteiramente pelo sistema ETL e não possuem fonte externa real. Essas são, geralmente, pequenas dimensões de pesquisa em que um código operacional é traduzido em palavras. Nesses casos, não há processamento real de Transform e Load. A pequena dimensão de pesquisa é simplesmente criada diretamente como uma tabela relacional em sua forma final. Mas o caso importante é a dimensão extraída de uma ou mais fontes externas.

Já descrevemos as quatro etapas do encadeamento do fluxo de dados de ETL em alguns detalhes. Aqui estão mais alguns pensamentos relacionados às dimensões especificamente.

Dados dimensionais para dimensões grandes e complexas, como cliente, fornecedor ou produto, são frequentemente extraídos de várias fontes em diferentes momentos. Isso requer atenção especial ao reconhecimento da mesma entidade dimensional em vários sistemas de origem, na resolução de conflitos em descrições sobrepostas e na introdução de atualizações nas entidades em vários pontos.

Tabelas de dimensão são tabelas planas desnormalizadas. Todas as hierarquias e estruturas normalizadas que podem estar presentes nas tabelas de preparo anteriores devem ser niveladas na etapa final de preparação da tabela de dimensões, se isso já não tiver ocorrido. A maioria dos atributos será de média e baixa cardinalidade. Por exemplo, o campo de gênero em uma dimensão de funcionário terá uma cardinalidade de três (masculino, feminino e não relatado) e o campo de estado em um endereço do Brasil terá uma cardinalidade de 27 (26 estados mais Distrito Federal, DF).

Se as tabelas de migração anteriores estiverem na terceira forma normal, essas tabelas de dimensão são facilmente produzidas com uma consulta simples na terceira fonte de formulário normal. Se todos os relacionamentos de dados apropriados forem impostos na etapa de limpeza de dados, esses relacionamentos serão preservados perfeitamente na tabela de dimensões planificada. No mundo da modelagem dimensional, a etapa de limpeza de dados é separada da etapa de entrega de dados, de forma que todos os relacionamentos de dados apropriados sejam entregues ao usuário final sem que o usuário precise navegar pelas estruturas normalizadas complexas.

É normal que uma dimensão complexa, como loja ou produto, tenha várias estruturas hierárquicas incorporadas simultâneas. Por exemplo, a dimensão de armazenamento pode ter uma hierarquia geográfica

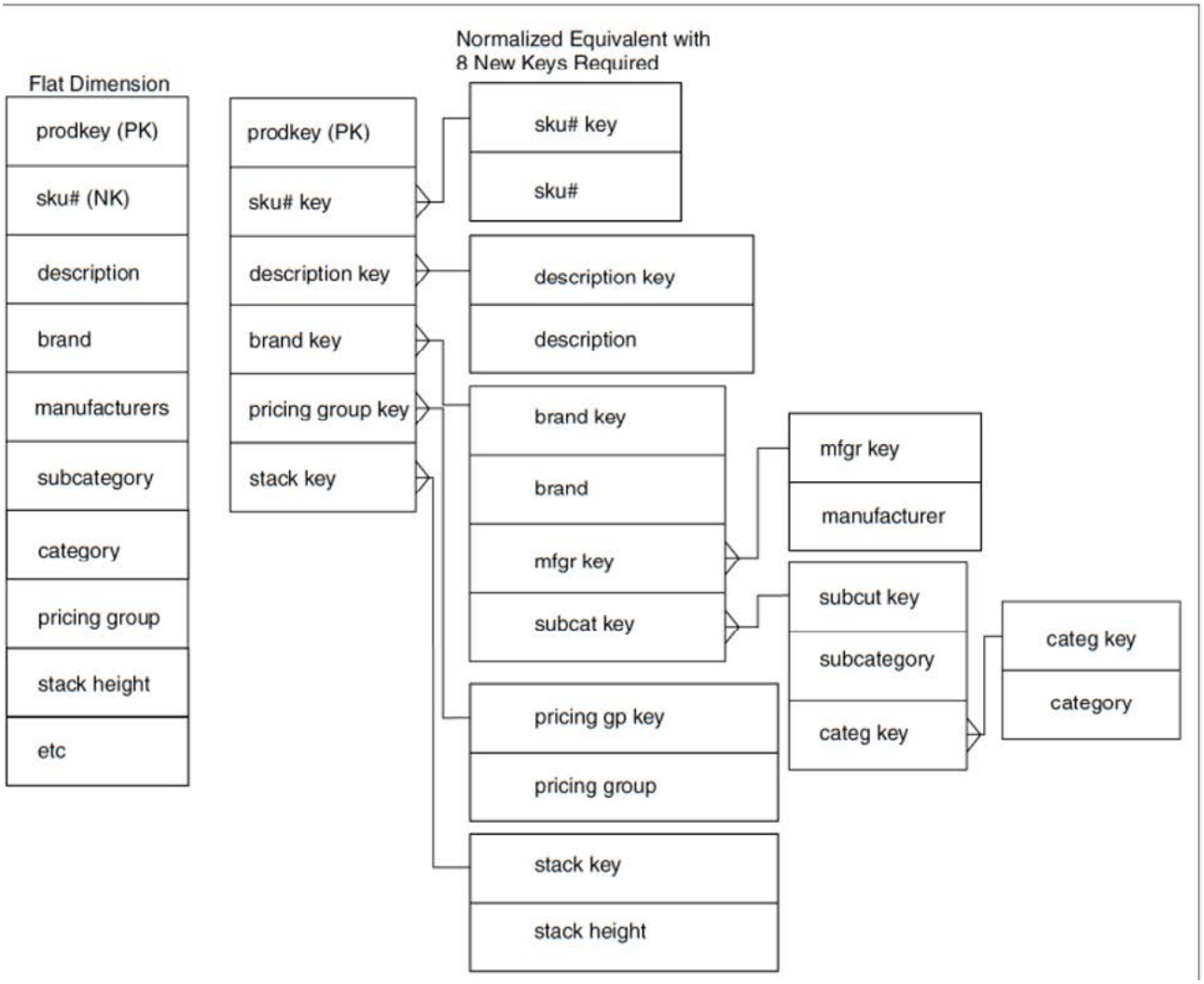
normal de localização, cidade, município e estado e também ter uma hierarquia de área de localização, distrito e região de merchandising. Essas duas hierarquias devem coexistir no mesmo repositório. Tudo o que é necessário é que cada atributo seja avaliado individualmente na presença da chave primária da tabela de dimensões.

1.1.1 Dimensões planas e dimensões de flocos de neve

Segundo Kimball e Caserta (2009), se uma dimensão é normalizada, as hierarquias criam uma estrutura característica conhecida como floco de neve, com os níveis das hierarquias com relacionamentos muitos-para-um perfeitos (Figura 16). É importante entender que não há diferença no conteúdo informacional entre as duas versões de dimensões nesta figura. A diferença com a qual nos importamos é o impacto negativo que o modelo normalizado e com flocos de neve tem no ambiente do usuário final. Existem dois problemas. Primeiro, se os muitos relacionamentos restritos em um modelo hierárquico mudarem, o esquema da tabela normalizada e as junções declaradas entre as tabelas devem ser alteradas, e o ambiente do usuário final deve ser recodificado em algum nível para que os aplicativos continuem funcionando. De acordo com Kimball e Caserta (2009), versões simples da dimensão não têm esse problema.

Segundo, os esquemas complexos são notórios por confundir os usuários finais, e um esquema normalizado requer o mascaramento dessa complexidade na área de apresentação do *data warehouse*. Geralmente, as tabelas de dimensão simples podem aparecer diretamente nas interfaces do usuário com menos confusão. Tendo criticado as dimensões do floco de neve, existem algumas situações em que é recomendado um tipo de neve. Estes são melhor descritos como subdimensões de outra dimensão. Se um atributo assumir diversos valores na presença da chave primária da dimensão, o atributo não poderá fazer parte da dimensão. Por exemplo, em uma dimensão de loja de varejo, o atributo de ID de caixa registradora assume muitos valores para cada loja.

Figura 16 – Versões planas e com flocos de neve de uma dimensão



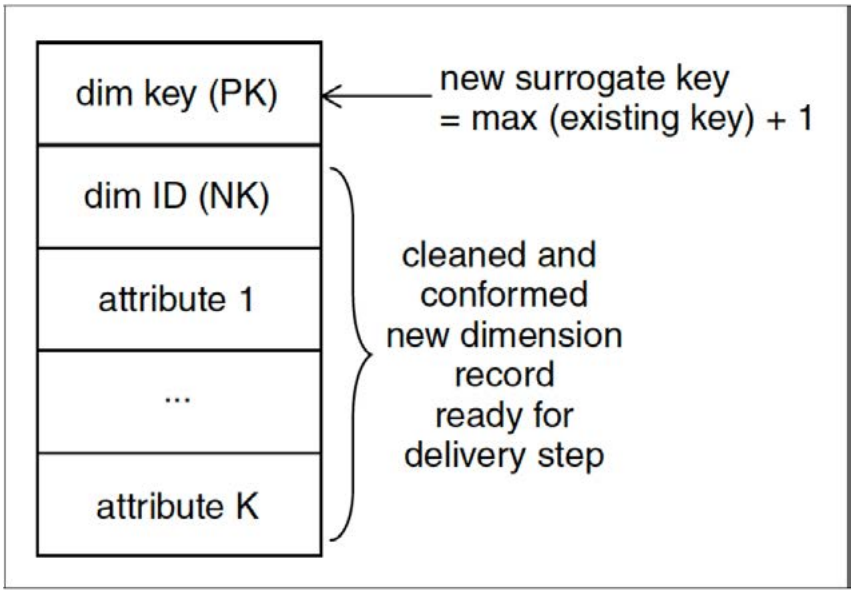
Fonte: KIMBALL; CASERTA, 2009, p. 169.

Cada vez que um novo registro de dimensão é criado, uma nova chave substituta deve ser atribuída. Veja a Figura 17. Esse inteiro sem sentido é a chave primária da dimensão. Em um ambiente de *data warehouse* centralizado, as chaves substitutas de todas as dimensões podem ser geradas a partir de uma única fonte. Nesse caso, um elemento de metadados mestre contém a chave mais alta usada para todas as dimensões simultaneamente.

No entanto, mesmo em um *data warehouse* altamente centralizado, se houver tarefas ETL simultâneas suficientes em execução, pode haver contenção para ler e gravar esse único elemento de metadados. E, claro, em um ambiente distribuído, essa abordagem não faz muito sentido.

Por esses motivos, recomenda-se que um contador de chave substituta seja estabelecido para cada tabela de dimensão separadamente. Não importa se duas chaves substitutas diferentes têm o mesmo valor numérico; o *data warehouse* nunca confundirá os domínios dimensionais separados, e nenhum aplicativo jamais analisa o valor de uma chave substituta, já que, por definição, não tem sentido.

Figura 17 – Atribuindo a chave substituta na etapa de dimensionamento



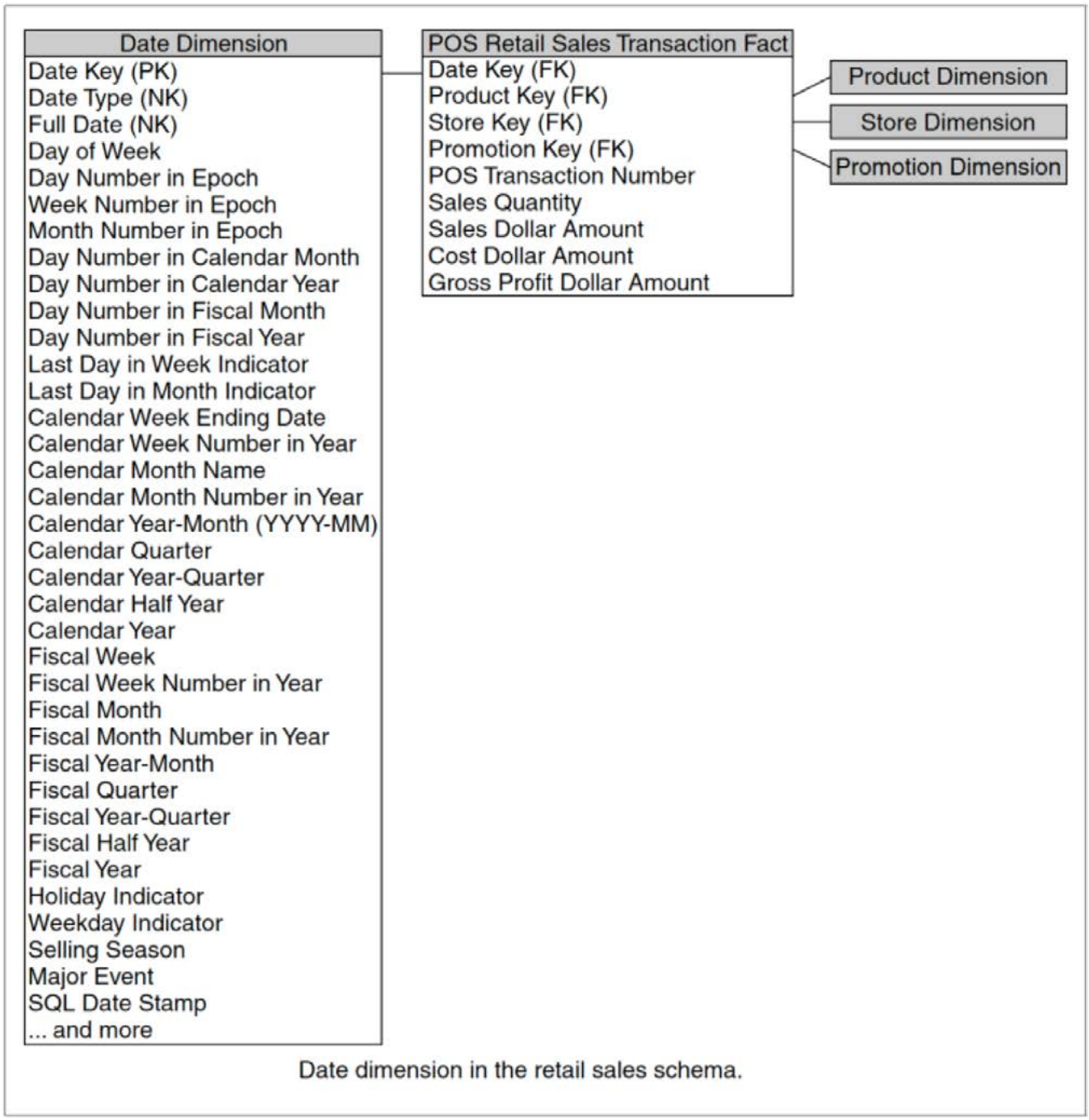
Fonte: KIMBALL; CASERTA, 2009, p. 170.

1.1.2 Dimensões de data e hora

Praticamente todas as tabelas de fatos possuem uma ou mais chaves estrangeiras (chaves primárias em outras tabelas) de dimensões relacionadas ao tempo. As medições são definidas em pontos específicos e a maioria das medições é repetida ao longo do tempo.
A dimensão de tempo mais comum e útil é a dimensão de data do calendário com a granularidade de um único dia. Essa dimensão tem, surpreendentemente, muitos atributos, como mostrado na Figura 18. Apenas alguns desses atributos (como nome do mês e ano) podem ser gerados diretamente de uma expressão de data e hora do SQL. Feriados, dias úteis, períodos fiscais, números de semanas, sinalizadores de último dia do mês e outros atributos de navegação devem ser incorporados na dimensão de data do calendário e toda a navegação de data deve ser implementada em aplicativos usando os atributos dimensionais.

A dimensão de data do calendário tem algumas propriedades muito incomuns. É uma das únicas dimensões completamente especificadas no início do projeto de *data warehouse*. Também não tem uma fonte convencional. A melhor maneira de gerar a dimensão de data do calendário é passar uma tarde com uma planilha e construí-la manualmente. Dez anos no valor de dias produzem menos que 4.000 linhas.

Figura 18 – Atributos necessários para uma dimensão de data do calendário



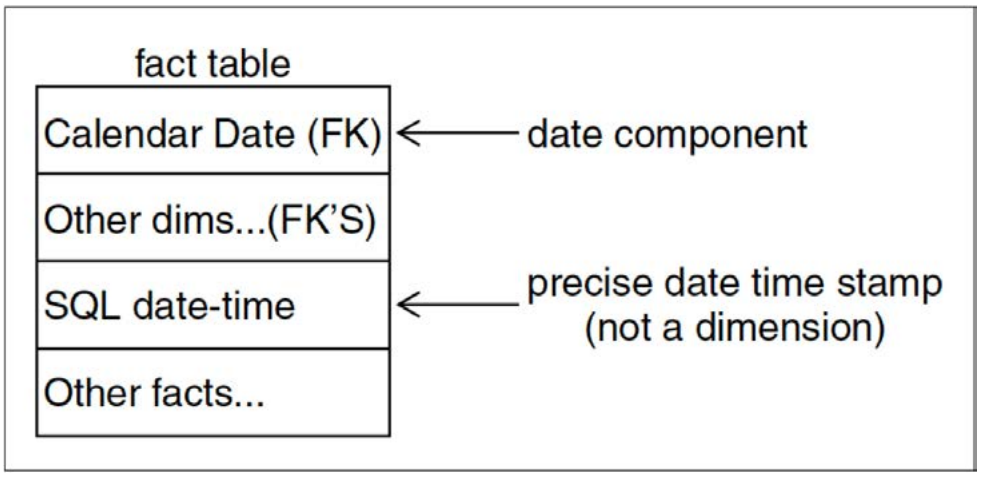
Fonte: KIMBALL; CASERTA. 2009, p. 171.

Cada dimensão de data do calendário precisa de um atributo de tipo de data e um atributo de descrição de data completa, conforme ilustrado na Figura 18. Esses dois campos compõem a chave natural da tabela. O atributo de tipo de data quase sempre tem a data do valor, mas deve haver pelo menos um registro que manipule a situação especial de data não aplicável, na qual a data registrada é inaplicável, corrompida ou ainda não aconteceu.

Em algumas tabelas de fatos, o tempo é medido em minutos ou até segundos. A intenção é preservar a poderosa dimensão de data do calendário e, ao mesmo tempo, oferecer suporte a consultas precisas no minuto ou no segundo. Também pode-se querer calcular intervalos de tempo muito precisos, comparando o tempo exato de dois registros da tabela de fatos. Por essas razões, recomenda-se o design mostrado na Figura 19.

O componente do dia do calendário do horário preciso permanece como uma referência de chave estrangeira à nossa dimensão familiar do dia do calendário. Mas também se incorpora um registro de data e hora completo de SQL diretamente na tabela de fatos para todas as consultas que requerem precisão extra. Nesse caso interessante, não é útil fazer uma dimensão com o componente de minutos ou segundos do registro de data e hora exato, porque o cálculo de intervalos de tempo entre os registros da tabela de fatos fica muito confuso ao tentar lidar com dia e hora do dia.

Figura 19 – Tabela de fatos para lidar com medições de tempo precisas



Fonte: KIMBALL; CASERTA, 2009, p. 173.

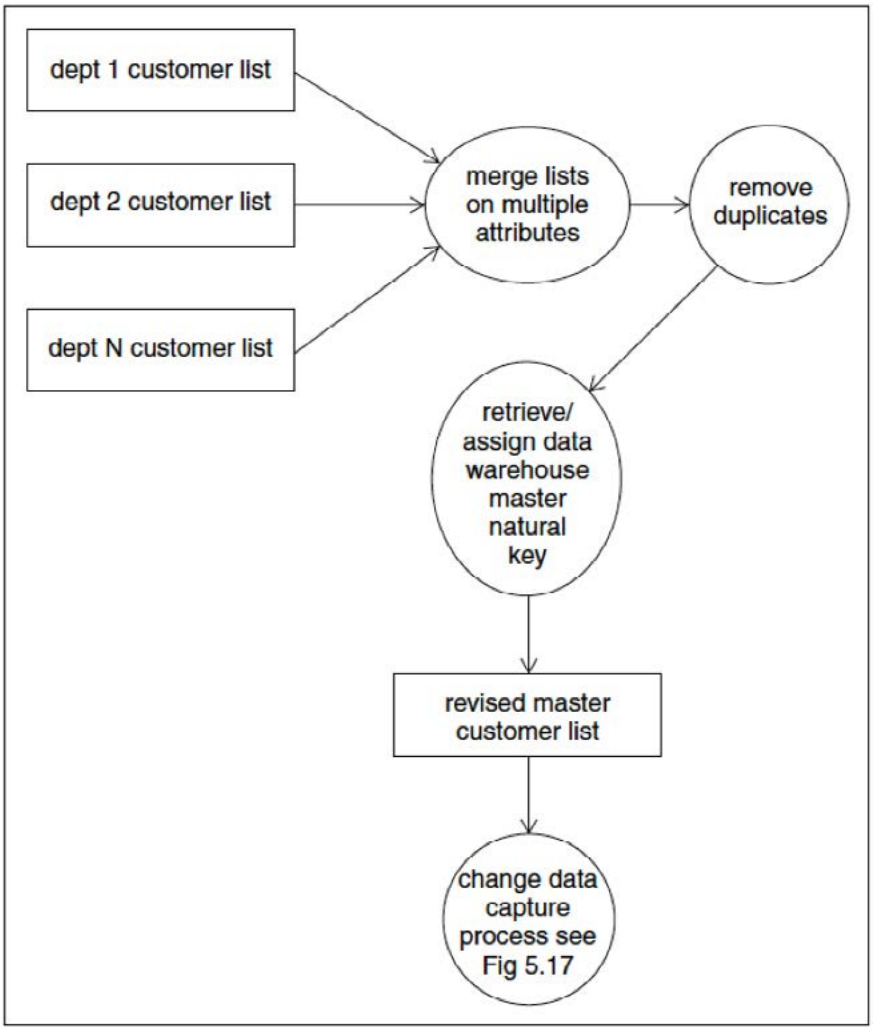
1.1.3 Dimensões grandes

De acordo com Kimball e Caserta (2009), as dimensões mais interessantes em um *data warehouse* são as grandes dimensões, como cliente, produto ou local. Uma grande dimensão de cliente comercial geralmente tem milhões de registros e cem ou mais campos em cada registro. Um grande registro de cliente individual pode ter dezenas de milhões de registros. Ocasionalmente, esses registros de clientes individuais têm dezenas de campos, mas, com mais frequência, essas dimensões monstruosas (por exemplo, clientes de mercearia identificados por um ID de comprador) têm apenas alguns atributos gerados por comportamento.

As dimensões realmente grandes quase sempre são derivadas de múltiplas fontes. Os clientes podem ser criados por um dos vários sistemas de gerenciamento de contas em uma grande empresa. Por exemplo, em um banco, um cliente poderia ser criado pelo departamento de hipoteca, pelo departamento de cartão de crédito ou pelo departamento de verificação e de poupança. Se o banco deseja criar uma única dimensão de cliente para uso por todos os departamentos, as listas de clientes originais separadas devem ter duplicatas removidas, conformadas e mescladas. Essas etapas são mostradas na Figura 20.

Na etapa de remoção de duplicatas, que faz parte do módulo de limpeza de dados, cada cliente deve ser identificado corretamente em fontes de dados originais separadas, para que a contagem total de clientes esteja correta. Uma chave mestra natural para o cliente pode ter que ser criada pelo *data warehouse* nesse ponto.

Figura 20 – Mesclar e remover duplicatas de vários conjuntos de clientes



Fonte: KIMBALL; CASERTA, 2009, p. 175.

PARA SABER MAIS

Na etapa de conformidade, que faz parte do módulo de conformidade com os dados, todos os atributos das fontes originais que tentam descrever o mesmo aspecto do cliente precisam ser convertidos em valores únicos usados por todos os departamentos. Por exemplo, um único conjunto de campos de endereço deve ser estabelecido para o cliente. Finalmente, na etapa de mesclagem (sobrevivência), que faz parte do módulo de entrega, todos os atributos separados restantes dos sistemas de origem individuais são unidos em um único registro de dimensão grande e amplo.

Dessa forma, você aprendeu as definições e os conceitos de conformação de dados. Você aprendeu a tratar problemas como redundância e conflito de valores, e técnicas para consolidação de dados.

TEORIA EM PRÁTICA

Uma empresa nacional de revenda de cosméticos está enfrentando alguns problemas financeiros. Dado o grande volume de produtos lançados pelo setor de cosméticos, a empresa está tendo dificuldades em acompanhar a demanda por produtos lançados, o que muitas vezes acaba por comprometer o resultado financeiro, por investir em produtos com baixa procura. Outro problema são os produtos que possuem prazo de validade curto, que acabam por vencer e não podem ser trocados pelos fornecedores. Hoje, a empresa conta com um sistema de vendas e controle de estoque, com banco de dados relacional e um processo de marketing pelas redes sociais, porém está tendo dificuldades para a tomada de decisões relacionadas ao que o seu público-alvo realmente consome e, assim, evitar gastos desnecessários. Após o processo de extração e limpeza e conformação dos dados do sistema de vendas e controle de dados e do marketing das redes sociais, os dados necessitam ser entregues. Como podemos auxiliar a organização no processo de entrega dos dados?

VERIFICAÇÃO DE LEITURA

1. A etapa final do processo de ETL é aquela em que são preparadas as estruturas de tabelas dimensionais de forma mais restrita. Assinale a alternativa que apresenta corretamente a etapa final do processo de ETL.

a. Extração dos dados.

b. Limpeza dos dados.

c. Conformação dos dados.

d. Entrega dos dados.

e. Exclusão dos dados.

2. O componente final de todas as dimensões, de acordo com Kimball e Caserta (2009), é um conjunto que vai além da chave primária e da chave natural. Assinale a alternativa que apresenta corretamente este conjunto.

a. Conjunto de atributos descritivos.

b. Conjunto de atributos chave primária.

c. Conjunto de atributos chave natural.

d. Conjunto de atributos de exemplificação.

e. Conjunto de atributos entregues.

3. De acordo com Kimball e Caserta (2009), as dimensões mais interessantes em um *data warehouse* são as dimensões, como cliente, produto ou local. Assinale a alternativa que apresente, corretamente, o tipo de dimensão ao qual o texto se refere:

a. Dimensões pequenas.

b. Dimensões grandes.

c. Dimensões de data.

d. Dimensões de hora.

e. Dimensões degeneradas.

Referências bibliográficas

KIMBALL, R.; CASERTA, J. **The Data Warehouse ETL Toolkit:** practical techniques for extracting, cleaning, conforming, and data delivering data. Indianopolis: Wiley Publishing, 2009.

Gabarito

Questão 1 – Resposta D

A etapa final do processo de ETL é a entrega de dados, em que são preparadas as estruturas de tabelas dimensionais de forma mais restrita.

Questão 2 – Resposta A

O componente final de todas as dimensões, além da chave primária e da chave natural, é o conjunto de atributos descritivos.

Questão 3 – Resposta B

De acordo com Kimball e Caserta (2009), as dimensões mais interessantes em um *data warehouse* são as grandes dimensões, como cliente, produto ou local.

Sistemas ETL de tempo real

Autor: Thiago Salhab Alves

Objetivos

- Compreender o que são sistemas ETL de tempo real.
- Conhecer as aplicações de ETL de tempo real.
- Conhecer as abordagens ETL de tempo real.

1. Introdução

Prezado aluno, você já parou para pensar em como as organizações, nacionais e multinacionais, realizam o processo de armazenamento de dados em tempo real após a extração, limpeza e conformação dos dados para compor um *data warehouse*? Diversas são as fontes de dados que apresentam dados heterogêneos e que necessitam ser combinados para compor uma base de dados unificada. Como adotar um processo de armazenamento de dados em tempo real (software que move dados para um *data warehouse* minutos após a execução da transação comercial)? Dessa forma, faz-se necessário conhecer o processo ETL de tempo real.

Esta leitura irá apresentar o que são sistemas ETL de tempo real e as aplicações e abordagens ETL de tempo real. Uma boa aula e bom trabalho.

2. Sistemas ETL de tempo real

De acordo com Kimball e Caserta (2009), ETL em tempo real é um nome impróprio para uma categoria de serviços de *data warehouse* que nem é verdadeiramente em tempo real nem, em muitos casos, é um processo de ETL. Em vez disso, o termo refere-se ao software que move os dados de maneira assíncrona para um *data warehouse* com certa urgência – minutos após a execução da transação comercial. Em muitos casos, a entrega de armazenamento de dados em tempo real exige uma abordagem bem diferente dos métodos ETL usados no *data warehouse* orientado para lotes. Simplesmente executar lotes ETL convencionais em um cronograma cada vez mais frequente ao longo do dia pode não ser prático, nem para os sistemas OLTP (Online Transaction Processing) nem para o *data warehouse*.

Segundo Kimball e Caserta (2009), construir uma solução ETL em tempo real exige que se classifiquem alguns objetivos comerciais, entendendo um conjunto diversificado de tecnologias, tendo consciência de algumas abordagens pragmáticas que foram empregadas com sucesso por outras pessoas e desenvolvendo flexibilidade e criatividade de engenharia. Este campo permanece jovem, com novas tecnologias, metodologias emergentes e novos vocabulários. Claramente, essa situação pode ser uma receita para problemas, mas o *data warehouse* em tempo real também oferece aos primeiros usuários um grande potencial para obter uma vantagem competitiva – um risco intrigante *versus* recompensa.

Não faz muito tempo, os engenheiros defenderam veementemente a noção de que o *data warehouse* precisava fornecer um conjunto inabalável de dados para os tomadores de decisão de negócios, fornecendo um piso confiável de informações sobre o qual se posicionar.

Para relatórios atualizados em um banco de dados, os usuários corporativos eram direcionados para os aplicativos de produção que executam os negócios. Portanto, os usuários precisavam ir até o *data warehouse* para ter uma visão histórica do que aconteceu no negócio a partir de ontem e precisavam examinar muitos sistemas OLTP (Online Transaction Processing) para obter uma imagem do que estava acontecendo hoje. Os usuários corporativos nunca aceitaram totalmente essa divisão. Por que eles não poderiam ir a um lugar único para obter as informações comerciais de que precisavam?

Embora o atraso entre uma transação comercial e sua disponibilização no *data warehouse* seja tipicamente inferior a 24 horas, para muitas organizações de rápida movimentação, esse atraso é muito grande.

Para atender à característica de tempo real, são necessárias estratégias diferentes das tradicionais para atualização do *data warehouse* (SANTOS; BERNARDINO, 2009). O ETL tradicional não está preparado para atualizações de baixa latência, sendo necessárias algumas adaptações para se atingir esse objetivo.

Segundo Kimball e Caserta (2009), vários outros fatores importantes conspiraram para forçar os profissionais de *data warehouse* a repensarem algumas posições anteriores:

- Gestão de relacionamento com clientes – CRM: o CRM moderno exige uma imagem contemporânea, consistente e completa do cliente disponível para todos os sistemas operacionais que, direta ou indiretamente, atendem o cliente – uma tarefa bastante difícil. Apesar das alegações de marketing dos principais fornecedores de CRM empacotados, esse recurso não pode ser adquirido da prateleira. A menos que todos os sistemas voltados para o cliente sejam retirados do pacote CRM empacotado, as empresas também precisam integrar informações do cliente em tempo real em todos os seus aplicativos transacionais herdados. Os *data warehouses*, obviamente, precisam de fluxos constantes de informações do cliente a partir das operações, mas, cada vez mais, os sistemas operacionais também contam com o enriquecimento do armazenamento de dados das informações do cliente. Portanto, é previsível que as organizações tenham começado a explorar alternativas arquiteturais que possam suportar cenários de integração mais generalizados, movendo dados operacionais entre aplicativos e, simultaneamente, dentro e fora do *warehouse* – com uma urgência cada vez maior.

- Negócios corporativos de latência zero ideal: este ideal exorta os benefícios da velocidade e uma única versão da verdade. Em uma empresa em tempo real, de latência zero, as informações são entregues no lugar certo, no momento certo, para obter o máximo valor comercial. Algumas pessoas chamam esses sistemas de tempo certo. Inventários *just-in-time* e cadeias de suprimento e modelos de negócios de customização em massa/montagem sob encomenda também ampliam a necessidade de informações absolutamente atuais e abrangentes em toda a organização. Atualmente, a latência zero real é um ideal inatingível – leva algum tempo para sincronizar informações entre vários sistemas de produção e *data marts* –, mas a pressão sobre muitos *data warehouses* modernos para fornecer uma visão de baixa latência da integridade dos negócios é muito real.

- Globalização e a web: por fim, e talvez de forma mais pragmática, os efeitos combinados da globalização e da web, que demandam operações e acesso 24 horas ao *data warehouse*, em conjunto com os requisitos para armazenar conjuntos de dados cada vez mais amplos e profundos, comprimiram seriamente a janela de tempo disponível para carregar o *data warehouse*. A quantidade de dados que precisa ser armazenada continua a se expandir, enquanto a janela de tempo de inatividade continua encolhendo, desafiando a equipe ETL de *data warehouse* já sobrecarregada. Não seria mais fácil se fosse possível, de alguma forma, alimentar seus *data warehouses* durante todo o dia em vez de tentar aumentar a carga de dados em janelas de tempo de inatividade aceitáveis?

O sistema OLTP (Online Transaction Processing) não tem o privilégio de esperar que a transação de carregamento do *data warehouse* seja confirmada antes de continuar com sua próxima transação, e nenhuma lógica de confirmação de bloqueio duas fases (*two-phase commit*) é prática em sistemas com diferentes estruturas e diferentes níveis de granularidade. Em vez disso, você deseja simplesmente mover as novas transações para uma partição especial em tempo real (definida posteriormente neste capítulo) do *data warehouse* dentro de um prazo aceitável para os negócios, fornecendo suporte analítico para as decisões operacionais do dia a dia. Por enquanto, esse procedimento é nossa definição prática de ETL em tempo real.

Segundo Kimball e Caserta (2009), o *data warehouse* em tempo real apresenta vários desafios e oportunidades exclusivos para o profissional de ETL. Os requisitos de disponibilidade do sistema podem ser escalados conforme a empresa confia na disponibilidade de baixa latência das transações de negócios no *data warehouse*. Se a organização optar pelas abordagens de gerenciamento de dimensão em tempo real descritas, a disponibilidade se tornará uma vantagem estratégica.

Do ponto de vista da arquitetura de dados, o armazenamento de dados em tempo real desafia a postura do *data warehouse* como sistema de medições periódicas discretas – um fornecedor de instantâneos de negócios – defendendo, em vez disso, um sistema de informações temporais mais abrangentes e contínuas. Essa mudança acontece sutilmente se, por exemplo, a frequência de carregamento de fatos aumenta de uma vez por dia para a cada 15 minutos, mas, mais drasticamente, se o carregamento de fatos e registros de dimensão ocorrer continuamente. O *data warehouse* pode, então, capturar um registro das transações comerciais e seu contexto dimensional em todos os pontos no tempo. Na verdade, se o *data warehouse* em tempo real também oferecer suporte à conformidade e sincronização de dimensão em tempo real, ele evoluirá para uma extensão lógica dos próprios sistemas operacionais.

A abordagem em tempo real para armazenamento de dados pode traçar uma linha clara para o que foi originalmente chamado de ADO (armazenamento de dados operacional). As motivações dos ADO originais eram semelhantes a dos *data warehouses* modernos em tempo real, mas a implementação de *data warehouses* em tempo real reflete uma nova geração de hardware, software e técnicas.

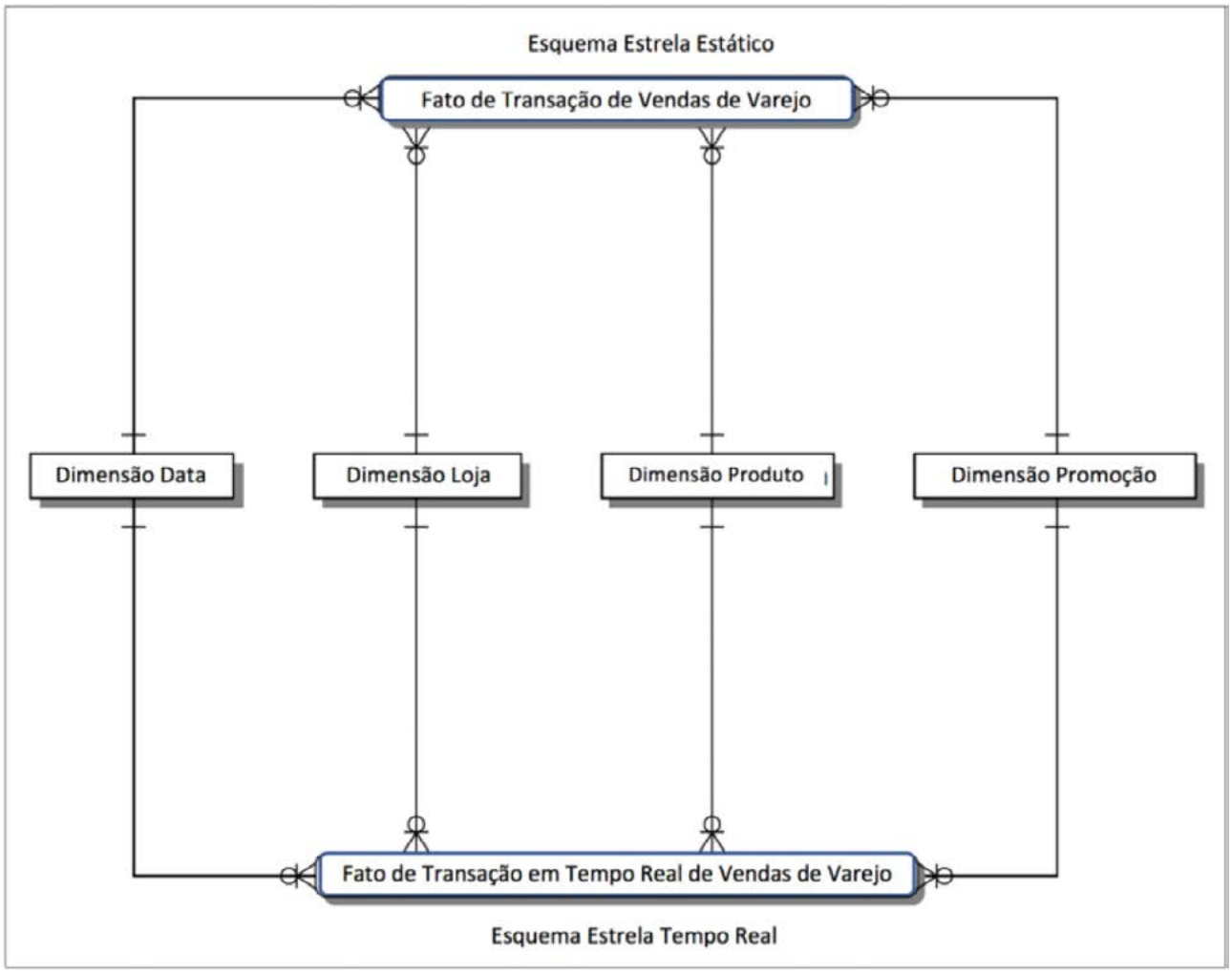
O armazenamento de dados operacionais, ou ADO, é uma construção de *data warehouse* de primeira geração, destinada a oferecer suporte a relatórios de baixa latência por meio da criação de uma construção arquitetônica distinta e de um aplicativo separado do *data warehouse*. O ADO é um sistema semioperacional e de apoio à decisão, tentando encontrar um equilíbrio entre a necessidade de suportar simultaneamente atualizações frequentes e consultas frequentes.

As primeiras arquiteturas ADO descreveram-no como um local onde os dados eram integrados e alimentados em um *data warehouse*, agindo, assim, como um tipo de extensão para a camada ETL do *data warehouse*. Arquiteturas posteriores retratam isso como um consumidor de dados integrados da camada de ETL do *data warehouse*, dependendo de onde a arquitetura geral reside e da urgência com a qual ela deve carregar dados do mundo operacional.

O uso da partição lógica e física em tempo real, conforme descrito originalmente por Ralph Kimball, é uma solução pragmática disponível para fornecer análises em tempo real a partir de um *data warehouse*. Usando essa abordagem, uma tabela de fatos em tempo real separada é criada, cuja granularidade e dimensionalidade correspondem à tabela de fatos correspondente no *data warehouse* estático (carregado por noite).

Essa tabela de fatos em tempo real contém apenas os fatos do dia atual (aqueles ainda não carregados na tabela do *data warehouse* estático). A Figura 21 mostra dois esquemas em estrela associados a uma tabela de fatos de ponto de venda de varejo em tempo real e estática, compartilhando um conjunto comum de dimensões.

Figura 21 – Relação entre esquema em estrela estático e tempo real

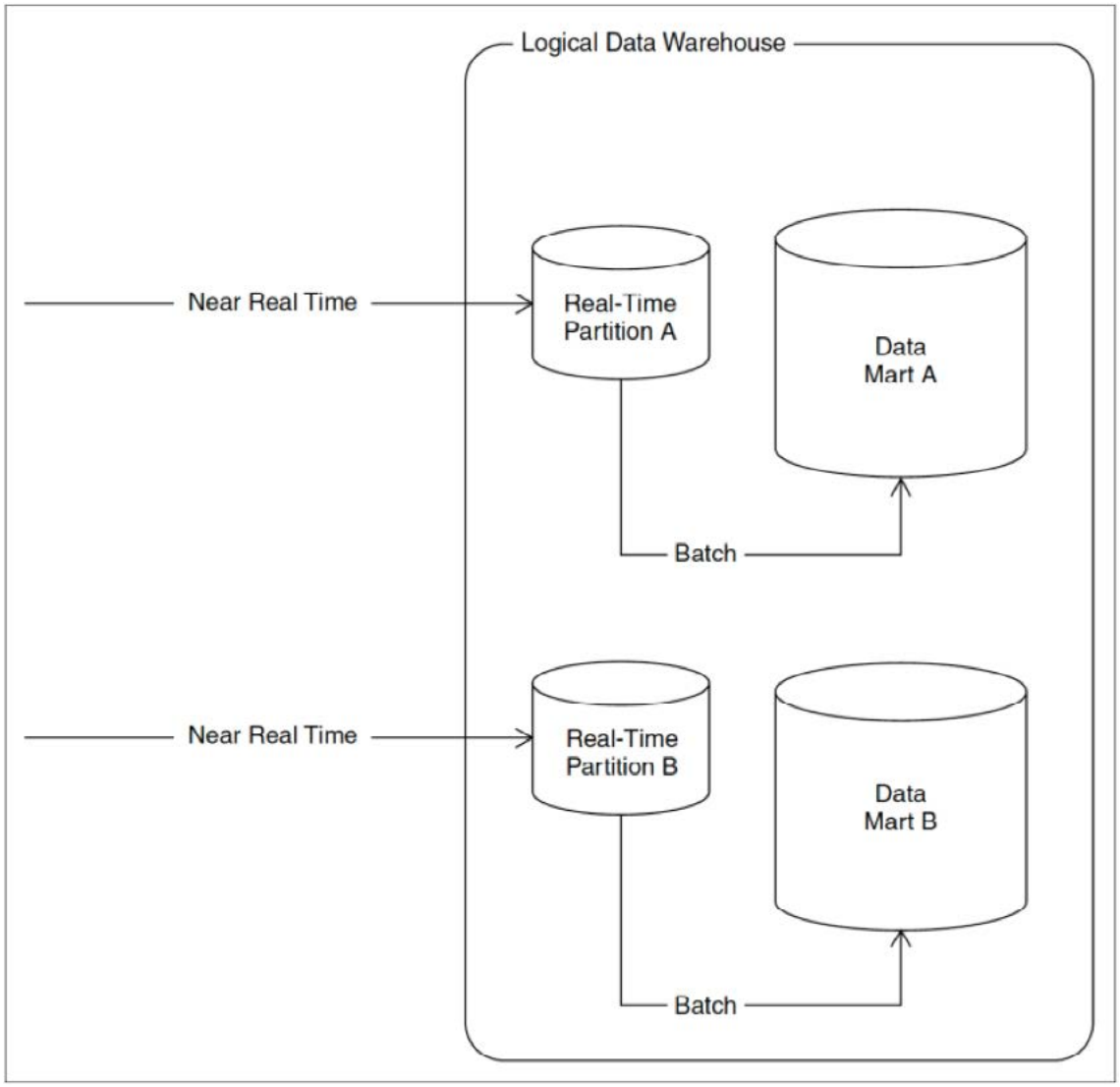


Fonte: adaptado de Kimball e Caserta (2009, p. 428).

A Figura 21 apresenta a relação entre esquemas em estrela estático e tempo real.

Todas as noites, o conteúdo da tabela de partições em tempo real é gravado na tabela de fatos estáticos e a partição em tempo real é então limpa, pronta para receber as transações do dia seguinte.
A Figura 22 dá uma ideia de como o processo funciona. Em essência, essa abordagem traz os benefícios de relatórios em tempo real do ODS para o próprio *data warehouse*, eliminando grande parte da sobrecarga arquitetural de ODS no processo.

Figura 22 – O relacionamento lógico da partição em tempo real para seu data mart



Fonte: KIMBALL; CASERTA, 2009, p. 429.

A Figura 22 apresenta relacionamento lógico da partição em tempo real para seu *data mart*.

Os fatos são transferidos para a(s) tabela(s) de fato em tempo real ao longo do dia, e as consultas do usuário na tabela em tempo real não são interrompidas por esse processo de carregamento. A indexação na tabela de fatos em tempo real é mínima ou inexistente para minimizar o esforço de carregamento de dados e seu impacto nos tempos de resposta da consulta. O desempenho é obtido restringindo a quantidade de dados na tabela (somente um dia) e armazenando em cache toda a tabela de fatos em tempo real na memória. Opcionalmente, é possível criar uma visualização (VIEW) que combina fatos (Uniões) na tabela de fatos estática e em tempo real, fornecendo um esquema em estrela virtual para simplificar as consultas que exigem visualizações de medidas históricas que se estenderam até o momento.

ASSIMILE

Segundo Kimball e Caserta (2009), os fornecedores de CRM estão bem cientes dos desafios enfrentados pelas organizações, portanto, alguns estão migrando os recursos de Business Intelligence para seus conjuntos de CRM operacionais. Com muita frequência, o resultado é rudimentar, simplista e difícil de defender arquitetonicamente, deixando de fornecer uma capacidade competitiva diferenciadora.

2.1 Abordagens ETL de tempo real

Segundo Kimball e Caserta (2009), o ETL convencional é extremamente eficaz para atender aos requisitos diários, semanais e mensais de relatórios em lote. Transações novas ou alteradas (registros de

fatos) são movidas em massa, e as dimensões são capturadas como instantâneos pontuais para cada carga. Assim, as alterações nas dimensões que ocorrem entre os processos em lote. O ETL, portanto, não é uma técnica adequada para integração de dados ou aplicativos para organizações que precisam de relatórios de baixa latência ou para organizações que precisam de uma captura de alterações dimensionais mais detalhada. Mas o ETL convencional é um método simples, direto e comprovado para organizações que possuem requisitos de latência e desafios complexos de integração.

2.2 Microbatch ETL

De acordo com Kimball e Caserta (2009), o Microbatch ETL é muito semelhante ao ETL convencional, exceto pelo fato de que a frequência de lotes é aumentada, talvez de hora em hora. Esses *microbatchs* frequentes são executados por meio de um processo ETL convencional e alimentam diretamente as partições em tempo real dos *data marts*. Uma vez por dia, as partições em tempo real são copiadas para os *data marts* estáticos e são esvaziadas.

O Microbatch ETL exige um método abrangente de controle de tarefas, cronograma, dependência e mitigação de erros, suficientemente robusto para ser executado sem supervisão durante a maior parte do tempo, e capaz de executar estratégias de publicação de *data warehouse* em face dos problemas mais comuns de carregamento de dados.

O Microbatch ETL também exige a detecção mais frequente de registros transacionais novos e atualizados nos sistemas OLTP, portanto, a carga imposta ao sistema operacional deve ser considerada e cuidadosamente gerenciada. Existem vários métodos para identificar os candidatos de registro alterados para carga de ETL do *microbatch* no *data warehouse* em tempo real:

- *Timestamps:* os registros de data e hora mantidos pelo sistema operacional para a criação e atualização de registros podem ser usados pelo ETL de *microbatch* em tempo real para diferenciar dados de candidatos para extração. Embora simples, esse método impõe gravações frequentes desses registros de data e hora nos sistemas operacionais para todas as alterações e leituras frequentes sempre que os processos ETL são executados. A indexação dos registros de data e hora melhora o desempenho de leitura e reduz a sobrecarga de leitura, mas aumenta a sobrecarga operacional em INSERTs e UPDATEs, às vezes de forma proibitiva. O profissional de ETL deve equilibrar essas preocupações.

- Tabelas de log ETL: outra abordagem é criar acionadores no ambiente OLTP para inserir os identificadores legados exclusivos de registros novos e alterados em uma série de tabelas especiais de log ETL. Essas tabelas especializadas existem apenas para acelerar o processamento de ETL e são usadas pelo processo de microblog ETL para determinar quais linhas foram alteradas desde o *microbatch* anterior. As tabelas de log ETL contêm o identificador exclusivo do registro dimensional novo ou alterado e talvez um valor de status, um registro de data e hora e um identificador de execução do processo ETL do *microbatch* que processa o registro alterado. O processo ETL do *microbatch* une as tabelas de log ETL às tabelas operacionais em que o identificador de execução ETL é nulo, extrai as linhas resultantes e, em seguida, exclui os registros de log ETL extraídos. A sobrecarga no sistema operacional é reduzida usando esse método, porque os INSERTs acionados por acionadores não exercitam indevidamente o sistema OLTP.

- Log dos sistemas gerenciadores de banco de dados: os arquivos de log de auditoria do SGBD, criados como subproduto de utilitários de *backup* e recuperação, às vezes podem ser utilizados para identificar transações novas e alteradas usando utilitários especializados chamados de *log-scraping*. Alguns desses utilitários de extração de logs podem, seletivamente, extrair e recriar

as instruções SQL aplicadas às tabelas de banco de dados de interesse desde algum ponto especificado no tempo, permitindo que o ETL saiba não apenas quais registros foram alterados desde a última extração, mas quais elementos foram alterados também nesses registros, informações que podem ser utilizadas pelo processo ETL ao aplicar diretamente as alterações nas tabelas de destino na área de preparação.

- Monitores de rede: esses utilitários monitoram algum conjunto de tráfego interessante em uma rede e filtram e registram o tráfego que eles veem. Os monitores de rede costumam ser usados para capturar o tráfego do fluxo da web, pois eliminam a necessidade de juntar os registros da web de vários servidores em um *farm* da web, fornecem sessões de visitas da web e melhoram a visibilidade do conteúdo real fornecido pelas páginas dinâmicas da web. Os monitores de rede são uma alternativa de ETL sempre que houver um fluxo de tráfego que exija análise de armazenamento de dados, incluindo roteamento de chamadas de telecomunicações e fluxo de trabalho de chão de fábrica.

2.3 Integração de aplicativos corporativos

De acordo com Kimball e Caserta (2009), no limite superior do espectro de complexidade está a integração de aplicativos corporativos (IAC), às vezes chamada de integração funcional. A IAC descreve o conjunto de tecnologias que suportam a verdadeira integração de aplicativos, permitindo que sistemas operacionais individuais interajam de maneiras novas e potencialmente diferentes do que foram originalmente projetados.

A IAC normalmente envolve a criação de um conjunto de componentes de adaptador e intermediário, que movimentam transações de negócios, na forma de mensagens, pelos vários sistemas na rede de integração, isolando todos os sistemas do conhecimento ou dependências de outros sistemas na rede de integração. Os

adaptadores específicos de aplicativos são responsáveis por lidar com toda a lógica necessária para criar e executar mensagens, e os agentes são responsáveis por rotear as mensagens adequadamente, com base nas regras de publicação e assinatura.

Adaptadores e corretores se comunicam por meio de mensagens independentes de aplicativos, geralmente em XML. Quando ocorre um evento de aplicativo significativo, como a atualização de um registro de cliente, um acionador é acionado e o adaptador do aplicativo cria uma nova mensagem. O adaptador também é responsável por iniciar as transações em seu respectivo aplicativo quando recebe uma mensagem contendo informações que ele escolheu para receber, como um registro de cliente recém-criado do sistema de gerenciamento de dimensão do cliente.

Os agentes encaminham mensagens entre adaptadores com base em um conjunto de regras de publicação e assinatura. As filas de mensagens são geralmente colocadas entre os aplicativos e seus adaptadores, e entre adaptadores e intermediários, para fornecer uma área temporária para mensagens assíncronas e para suportar garantias de entrega e consistência de transações na rede de integração.

As tecnologias IAC (integração de aplicativos corporativos) podem ser poderosas, habilitando ferramentas para o *data warehouse* em tempo real, pois suportam a capacidade de sincronizar dados importantes, como informações do cliente em aplicativos, e fornecem meios eficazes para distribuir ativos de informações derivadas de *data warehouse*, como novos valores de segmentação de clientes, em toda a empresa.

A arquitetura de *data warehouse* IAC em tempo real modulariza o bloco monolítico de ETL, tornando os sistemas do gerenciador de dimensão como componentes arquitetônicos separados, cada um com seus próprios adaptadores, e responsabilizando-se pela maioria das tarefas de transformação e carregamento do *data mart* real.

Um cenário real típico pode envolver a implementação de adaptadores para um conjunto de sistemas OLTP, como planejamento de recursos corporativos, ERP e automação da força de vendas, sistemas de gerenciamento de dimensão de cliente e produto (que executam limpeza e desduplicação em tempo real) e *data marts* para pedidos e chamadas de vendas.

2.4 Capturar, transformar e fluxo

Segundo Kimball e Caserta (2009), o Capture, Transform e Flow (CTF) é uma categoria relativamente nova de ferramentas de integração de dados, projetada para simplificar o movimento de dados em tempo real por meio de tecnologias de bancos de dados heterogêneos. A camada de aplicativo dos aplicativos transacionais é ignorada. Em vez disso, as trocas diretas do banco de dados com o banco de dados são executadas. Transações, tanto novos fatos quanto mudanças de dimensão, podem ser movidos diretamente dos sistemas operacionais para as tabelas intermediárias de *data warehouse* com baixa latência, normalmente alguns segundos.

A funcionalidade de transformação das ferramentas CTF é normalmente básica em comparação com as ferramentas de ETL maduras de hoje, as soluções CTF de *data warehouse* em tempo real envolvem a movimentação de dados do ambiente operacional, transformando-a levemente usando a ferramenta CTF. Essas tarefas leves de transformação podem incluir a padronização de formatos de data, a reformulação de tipos de dados, o truncamento ou a extensão de campos. Depois que os dados são testados, transformações adicionais além das capacidades da ferramenta CTF são aplicadas conforme necessário. Essas transformações subsequentes podem ser chamadas pelo ETL do *microbatch* ou por disparadores que são acionados no INSERT na área de preparação.

Em qualquer cenário de transformação, os registros são gravados diretamente nas tabelas de partição em tempo real do *data mart*. Essas transformações subsequentes podem incluir tarefas como validação de

dados, limpeza e correspondência de registros de dimensões, pesquisas de chave substitutas para registros dimensionais e criação de novos registros dimensionais de alteração lenta, conforme necessário.

PARA SABER MAIS

De acordo com Kimball e Caserta (2009), o Microbatch ETL é uma excelente opção para requisitos de *data warehouse* tolerantes a latência horária sem atualizações dimensionais intra-hora e que não exigem sincronização bidirecional de dados dimensionais entre o *data warehouse* e os sistemas operacionais. É, de longe, a abordagem mais simples para entregar relatórios de *data warehouse* quase em tempo real.

Dessa forma, você aprendeu o que são sistemas ETL de tempo real e as abordagens ETL de tempo real.

TEORIA EM PRÁTICA

Uma empresa nacional de revenda de cosméticos está enfrentando alguns problemas financeiros. Dado o grande volume de produtos lançados pelo setor de cosméticos, a empresa está tendo dificuldades em acompanhar a demanda por produtos lançados, o que muitas vezes acaba por comprometer o resultado financeiro, por investir em produtos com baixa procura. Hoje, a empresa conta com um sistema de vendas e controle de estoque, com banco de dados relacional e um processo de marketing pelas redes sociais, porém está tendo dificuldades para a tomada de decisões relacionadas ao que o seu

público-alvo realmente consome e, assim, evitar gastos desnecessários. Após o processo de extração e limpeza, conformação e entrega dos dados do sistema de vendas e controle de dados e do marketing das redes sociais, constatou-se que os dados deveriam ser armazenados em tempo real. Como podemos auxiliar a organização no processo de armazenamento em tempo real?

VERIFICAÇÃO DE LEITURA

1. Para atender à característica de tempo real, são necessárias estratégias diferentes das tradicionais para atualização do *data warehouse*. O ETL tradicional não está preparado para realizar que tipo de atualizações?

   a. Baixa latência.

   b. Alta latência.

   c. Dados heterogêneos.

   d. Dados de diferentes tipos.

   e. Dados de diferentes tamanhos.

2. ETL em tempo real é um nome impróprio para uma categoria de serviços de *data warehouse* que nem é verdadeira em tempo real nem, em muitos casos, ETL. O termo refere-se ao software que move os dados de que maneira para um *data warehouse*?

a. Assíncrona.

b. Síncrona.

c. Homogênea.

d. Heterogênea.

e. Tempo real.

3. É uma construção de *data warehouse* de primeira geração destinada a oferecer suporte a relatórios de baixa latência por meio da criação de uma construção arquitetônica distinta e de um aplicativo separado do *data warehouse*. Assinale a alternativa que apresente, corretamente, o tipo de construção ao qual o texto se refere:

   a. Armazenamento de dados operacional.

   b. Limpeza de dados operacional.

   c. Extração de dados operacional.

   d. Transformação de dados operacional.

   e. Exclusão de dados operacional.

Referências bibliográficas

KIMBALL, R.; CASERTA, J. **The Data Warehouse ETL Toolkit:** practical techniques for extracting, cleaning, conforming, and data delivering data. Indianopolis: Wiley Publishing, 2009.

SANTOS R. J.; BERNARDINO J., 2009. Optimizing data warehouse loading procedures for enabling useful-time data warehousing. In: International Database Engineering & Applications Symposium (IDEAS '09), New York. **Proceedings...** New York, USA, ACM, 2009. p. 292- 299.

Gabarito

Questão 1 – Resposta A

O ETL tradicional não está preparado para realizar atualizações de baixa latência, sendo necessárias algumas adaptações para atingir esse objetivo.

Questão 2 – Resposta A

O termo refere-se ao software que move os dados de maneira assíncrona para um *data warehouse* com certa urgência.

Questão 3 – Resposta A

Armazenamento de dados operacional é uma construção de *data warehouse* de primeira geração, destinada a oferecer suporte a relatórios de baixa latência por meio da criação de uma construção arquitetônica distinta e de um aplicativo separado do *data warehouse*.

Bons estudos!