DOI: 10.5748/9788599693124-13CONTECSI/PS-4235

STUDIES OF METHODS AND PROCESS REFACTORING DATABASE IN LITERATURE

Jair Moratore Junior (Centro Universitário Eurípides de Marília, São Paulo, Brasil) - juninho_ghost92@hotmail.com
Elvis Fusco (Centro Universitário Eurípides de Marília, São Paulo, Brasil) - elvisfusco@univem.edu.br

This study presents refactoring techniques in relational database aimed at correcting errors and anomalies that arise over time due to imperfections and flaws in the conceptual schema of the database, due to the inefficiency of the data modeling process which leads to duplication information, inconsistencies, among other problems, leaving the less reliable database and their increasingly difficult information to be used in decisionmaking processes. It is proposed the application of database refactoring techniques of structural categories, data quality, referential integrity and architectural in case of use, which tends to minimize errors and facilitate understanding of the information contained in the database, thus also facilitates the consultation, keeps the semantics and improves its structure, preserving its original features without adding new ones, improving schemes for future projects in the database. Finally, it is performed a comparison with the database refactoring techniques described in the literature.

Keywords: Database, Refactoring, Database Refactoring.

ESTUDO DE MÉTODOS E PROCESSOS DE REFATORAÇÃO DE BANCO DE DADOS NA LITERATURA

Este estudo apresenta técnicas de refatoração em banco de dados relacional que visam corrigir erros e anomalias que surgem com o passar do tempo, devido a imperfeições e falhas nos esquemas conceituais do banco de dados, fruto da ineficiência do processo de modelagem de dados que acarretam duplicações de informações, inconsistências, entre outros problemas, deixando o banco de dados menos confiável e suas informações cada vez mais difíceis de serem utilizadas no processos de tomada de decisão. Propõese a aplicação das técnicas de refatoração de banco de dados das categorias estrutural, qualidade de dados, integridade referencial e arquitetural em casos de uso, que tende a minimizar erros e facilitar o entendimento das informações contidas no banco de dados, assim facilitar consultas, manter a semântica e melhorar sua estrutura, conservando suas funcionalidades originais sem adicionar novas funcionalidades, melhorando os esquemas para projetos futuros em banco de dados. Ao final, é realizada uma comparação com as técnicas de refatoração de banco de dados descritas na literatura.

Palavras-chaves: Banco de dados, Refatoração, Refatoração de Banco de Dados.

INTRODUÇÃO

O banco de dados é um bem vital para qualquer organização, sendo a mesma de pequeno, médio ou principalmente de grande porte. Nesse ambiente todos os dados e informações não apenas da organização, mas também de pesquisas de mercado e faturamento estão armazenadas no banco de dados. Todas as organizações de modo geral se preocupam com a segurança de suas respectivas informações, sem levar em conta que uma boa modelagem conceitual no esquema lógico do banco é um modo de segurança preventiva para o futuro banco de dados que estará funcionando dentro da organização.

Um projeto de banco de dados contém mais do que futuras tabelas e relacionamentos entre elas, contém cenário, modelagem lógica, modelagem física, tudo para que o banco de dados a ser instalado não contenha erros como: redundâncias de informações, duplicações, dados isolados, anomalias, dentre outros, imperfeições que de algum modo afetam a confiabilidade, semântica e integridade do banco de dados e de suas informações.

As técnicas de refatoração podem ajudar a minimizar esses erros, que por algum motivo passam despercebidos, pode facilitar nas consultas a dados, e melhorar os esquemas conceituais do banco, deixar o projeto mais confiável mantendo a semântica e integridade não apenas dos dados, mas também do banco de dados em questão.

A refatoração tende a melhorar os esquemas do banco a partir do modelo lógico, onde os erros são tratados com menor frequência, esta pesquisa toma como premissa, a utilização de banco de dados relacionais. Este artigo tem como objetivo estudar e aplicar as técnicas de refatoração em banco de dados relacional para o melhor aproveitamento dos dados no banco, corrigindo erros e redundâncias a partir do modelo lógico do banco de dados.

A partir da revisão bibliográfica sobre modelagem de dados, refatoração, refatoração de banco de dados e metodologias ágeis foi feita a verificação da aplicação dos conceitos de refatoração e metodologias ágeis no processo de refatoração de banco de dados e a utilização de um banco de dados legado para aplicação das técnicas de refatoração de banco de dados selecionadas, para atingir ao objetivo proposto. A discussão inicia-se conceituando-se banco de dados e detalhando a sua implantação lógica. Na sequencia apresentam-se considerações sobre refatoração e refatoração de banco de dados, para, finalmente, apresentar um estudo comparativo de técnicas de refatoração de banco de dados descritos na literatura.

1 IMPLANTAÇÃO LÓGICA DO BANCO DE DADOS

O Banco de Dados (BD) é uma representação simples, normalmente gráfica, de estrutura de dados reais mais complexas, sendo esta um modelo de uma abstração de um objeto ou evento real de maior complexidade. Sua função é auxiliar na compreensão das complexidades do ambiente real. Pela visão do BD, nada mais é que um modelo que representa estruturas de dados com suas características, relações, restrições, transformações e outros elementos que tenham finalidade de dar suporte ao problema específico de um domínio (Macedo, 2011).

Yong (1990) enfatiza a importância do modelo de dados principalmente:

- Na possibilidade de fornecer informações sobre todos os dados envolvidos e de forma global, dando oportunidade ao usuário de conhecer a semântica do banco de dados.
- A possibilidade adicional de conhecer a estrutura das informações das condições de restrição quanto à integridade, de segurança e privacidade do banco de dados.

Yong (1990) conceitua que em um modelo de dados devem-se considerar alguns critérios para sua implantação e construção que são:

- O modelo deve ser simples, isto é, conter o menor número possível de elementos referentes á estrutura de dados.
- O modelo deve ser exibido com boa visualização, por meio de métodos e técnicas descritivas facilitando a compreensão do modelo.
- O modelo deve conter elementos que sejam contrapartidas diretas dos conceitos básicos do mundo real percebido.
- O modelo deve conter elementos tanto de estrutura como de substancia, que sirvam para a construção fácil de esquemas.
- O modelo deve conter o mínimo possível de atributos típicos da fase de implantação, isto é, seja independente de SGBD, no entanto não deve ser muito distante da possível estrutura física, a fim de diminuir a complexidade de transformação do modelo lógico ao físico.
- O modelo deve conter indicações para a formação de partições de dados, com finalidades de controle de acessos ao banco de dados, assim como maior eficiência de pesquisa online e controle de redundâncias.

Yong (1990) afirma que um modelo de dados é constituído de duas fases, que são: Conceituação do modelo e implantação do modelo de dados, e que nas mesmas estão contidos quatro etapas que são: percepção do mundo real; representação do mundo real, através do modelo conceitual; implantação lógica e implantação física. Para os propósitos deste estudo detalharemos a implantação lógica.

Com referência a modelo de dados, a fase de implantação lógica pode ser considerada como sendo análise de rotas lógicas de acesso dos diversos modelos de dados.

Tais rotas de acesso tem maior facilidade de aplicação nos modelos de rede e hierarquia, dada à facilidade de visualização via diagramas de estruturas de dados (Yong, 1990).

Os outros modelos, mais próximos do nível conceitual, metal (por exemplo, relacional, e entidade relacionamento), não efetuam análise de rotas de acesso, nem pela finalidade de utilização de modelo (o modelo entidade-relacionamento é um modelo típico para modelagem e inadequado a análises de rotas) nem como técnicas de manipulação especiais (o modelo relacional utiliza operadores e álgebra relacional na composição, em tempo de consulta, “rotas de acesso”) (Yong, 1990).

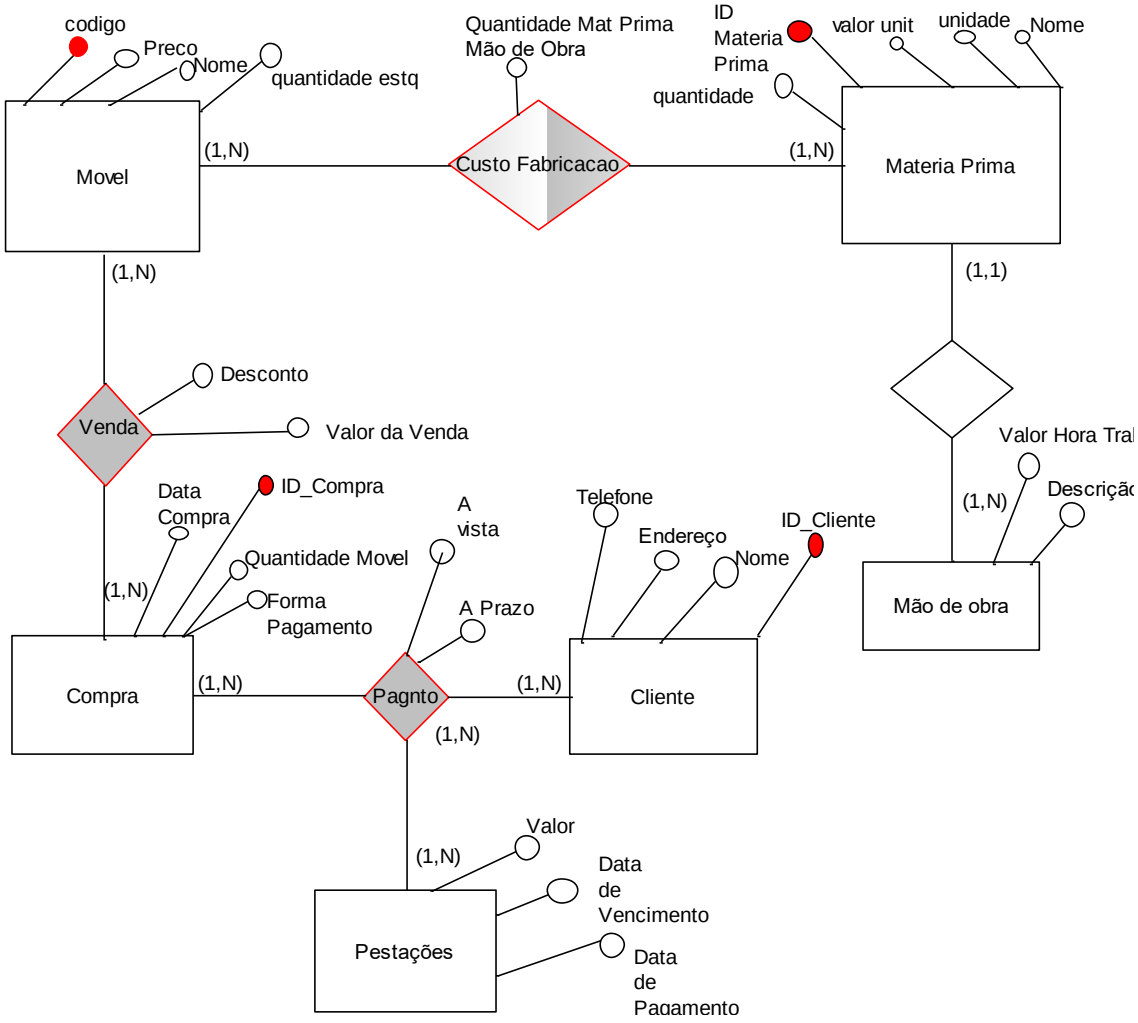
Segundo Chen (1990) o projeto lógico de banco de dados é o processo de planejar a estrutura lógica de dados para o banco de dados, isto envolve uma análise do ambiente de aplicação e dois tipos de estruturas lógicas de dados disponíveis no sistema do banco de dados.

O modelo lógico já leva em conta algumas limitações e executa recursos como adequação de padrão e nomenclatura, define as chaves primárias e estrangeiras, normalização, integridade referencial, entre outras. O modelo lógico deve ser criado de acordo com os exemplos de modelagem de dados criados no modelo conceitual.

A Figura 1 ilustra o modelo lógico de dados, o diagrama E-R (entidade-

relacionamento) de uma fábrica de móveis, com suas respectivas entidades, cardinalidades mínimas e máximas de cada relação, e chaves primárias.

Figura 1. Modelo Lógico de Dados



Fonte: Autoria própria

Segundo Chen (1990) o modelo Entidade-Relacionamento para projeto lógico de banco de dados consiste nas seguintes etapas:

1. Identificar tipos de entidades.
2. Identificar tipos de relacionamentos.
3. Desenhar um diagrama E-R com tipos de entidade e relacionamentos.
4. Identificar tipos de valores e atributos.
5. Traduzir o diagrama E-R em um diagrama de estrutura de dados.
6. Projetar formatos de registros.

Chen (1990) relata alguns problemas no modelo ou projeto lógico de banco de dados.

1) O projetista de banco de dados tem de considerar muitas questões ao mesmo tempo, o que torna a tarefa de projeto de banco de dados muito difícil.

2) O resultado final do projeto lógico de banco de dados é o esquema (ou seja, uma descrição da visão do banco de dados pelo usuário). Uma vez que o esquema do usuário representa a solução do projetista para as questões complicadas.

Visando minimizar esses problemas, apresenta-se a seguir o conceito de refatoração, bem como o melhoramento da refatoração em um projeto de software; como fazer a refatoração e ajudar a encontrar falhas e também os problemas com a mesma.

2 REFATORAÇÃO

Fowler et. al. (1999: 50-51) define refatoração em dois termos, em um substantivo e um verbo, apresentados a seguir

> Refatoração (substantivo) uma alteração feita na estrutura interna do software para torna-lo mais fácil de ser entendido e menos custoso de ser modificado sem alterar seu comportamental observável.
> Refatoração (verbo) reestruturar software aplicando uma serie de refatorações sem alterar seu comportamento observável.

Refatoração fornece uma técnica para limpar código de uma maneira mais eficiente e controlada (Fowler et. al., 1999).

Sem refatoração, o projeto do programa acaba por se deteriorar. A medida com que as pessoas alteram o código, alterações para executar objetivos de curto prazo ou então alterações feitas sem uma compreensão total do projeto do código, o código se desestrutura (Fowler et. al., 1999).

Torna-se mais difícil visualizar o projeto ao ler o código. Refatoração é como arrumar o código. As partes do código que não estejam no lugar apropriado são removidas, a perda de estrutura do código tem um efeito cumulativo, quanto mais difícil é visualizar o projeto a partir do código, mais difícil será preservá-lo e mais rapidamente ele se desestruturará, a refatoração sistemática ajuda o código a conservar a sua forma (Fowler et. al., 1999).

Um sistema mal projetado normalmente precisa de mais código para fazer as mesmas coisas, muitas vezes porque o mesmo código é replicado em diversos lugares diferentes, assim, um aspecto importante na melhoria do projeto é a eliminação de código duplicado, a importância disto recai sobre futuras modificações no código, reduzir a quantidade de código não fará o sistema rodar mais rapidamente, porque o efeito no desempenho dos programas é raramente significativo, reduzir a quantidade de código faz, todavia, uma grande diferença sobre a manutenção deste código, quanto mais código houver, mais difícil será modificá-lo corretamente (Fowler et. al., 1999).

Refatorar pode ajudar a encontrar falhas no código, que não foram encontradas no início do desenvolvimento, o objetivo aqui é entender o código, o que o código faz, e sendo encontradas falhas, corrigi-las, para um código mais confiável e com menos erros e falhas possível.

Segundo Fowler et. al. (1999: 53)

> A melhora no entendimento do código também me ajuda a localizar falhas. Admito que não sou muito bom em encontrar falhas, algumas pessoas conseguem ler um bloco de código e achar falhas, eu não, entretanto descubro que, se recapturo código, trabalho pesado na compreensão do que ele faz, e aplico esse novo conhecimento de volta no código.

Refatorar ajuda você a desenvolver código mais rapidamente. As pessoas podem ver facilmente que ela melhora a qualidade. Melhorar o projeto, melhorar a legibilidade, reduzir falhas, tudo isto melhora a qualidade. Refatorar exige tempo, mas, com o desenvolvimento em andamento separar tempo para refatorar pode deixar o desenvolvimento mais demorado, refatoração não deve ser feita de uma vez ou em certo período do desenvolvimento mas, a todo tempo como afirma Fowler et. al. (1999).

No conserto de falhas, muito da utilidade da refatoração vem do fato de tornar o código mais compreensível. Segundo Fowler et. al. (1999: 55)

> Quando olho o código tentando compreendê-lo, refatoro para ajudar a melhorar minha compreensão. Muitas vezes descubro que este processo ativo de trabalho com o código ajuda a encontrar falha. Um modo de encarar isso é que, se você recebe uma notificação de falha, é um sinal de que precisa refatorar, porque o código não estava claro o suficiente para que percebesse que havia uma falha.

Segundo Fowler et. al. (1999) algumas organizações fazem revisões de código regulares. Aquelas que não o fazem seriam melhores se fizessem. Revisões de código ajudam a difundir conhecimento através de uma equipe de desenvolvimento. As revisões facilitam a transmissão do conhecimento dos desenvolvedores mais experientes para os menos experientes. Elas ajudam as pessoas a entender melhor os aspectos de um sistema grande de software. Elas também são muito importantes na escrita de código.

Segundo Fowler et. al. (1999) uma área problemática para refatoração são os bancos de dados. A maioria das aplicações comerciais são altamente acopladas ao esquema de banco de dados que as suportam. Esse é o motivo pelo qual os bancos de dados são difíceis de modificar. Outro motivo é a migração de dados.

Mesmo se você tiver colocado seu sistema cuidadosamente em camadas para minimizar as dependências entro esquema do banco de dados lhe força a migrar os dados, o que pode ser uma tarefa longa e pesada.

3 REFATORAÇÃO DE BANCO DE DADOS

Segundo Ambler e Sadalage Apud Domingues (2014) uma refatoração em banco de dados pode ser definida como uma alteração simples no seu esquema, com o objetivo de melhorar seu projeto, preservando sua semântica informacional e sua semântica comportamental, assim para se realizar uma refatoração não é possível adicionar novas funcionalidades ou modificar alguma existente, não se pode alterar um dado e nem alterar o significado de um dado existente (Domingues, 2014).

A refatoração de um banco de dados consiste em uma melhoria junto a sua estrutura, onde se aprimora o projeto sem a adição de novas funcionalidades e sem alteração das funcionalidades já existentes ou da semântica informacional (Dalpra e Araújo, 2012).

Refatorar um banco de dados introduz uma fase intermediária necessária entre o ajuste e as correções, onde se propõe a realização dos processos de maneira diferente com ganho significativo em eficiência, em comparação com a mesma aplicação no passado (Dalpra e Araújo, 2012).

Há uma diferença crucial entre refatoração e adição de nova funcionalidade. Na refatoração busca-se uma melhora da estrutura existente, já ao inserir uma nova funcionalidade acrescenta-se uma nova estrutura. Em alguns casos pode ser necessário

refatorar o banco de dados para posteriormente inserir uma nova funcionalidade, ou então se pode identificar à necessidade de refatorar a nova funcionalidade inserida junto ao banco. Sendo assim, conclui-se que refatoração e adição de novas funcionalidades são coisas distintas, porém complementares (Dalpra e Araújo, 2012).

Em outras palavras, após a execução de uma refatoração ou um conjunto de refatorações no esquema de dados, o banco de dados deve responder as mesmas consultas que eram realizadas antes deste processo de refatoração (Domingues, 2014).

A estratégia da refatoração definida por Ambler é aplicada em ambientes complexos, nos quais existem várias aplicações sendo acessadas simultaneamente no mesmo banco de dados. Quando esse banco precisa ser alterado, praticamente todas as aplicações precisam sofrer algum tipo de adaptação, no entanto, não é possível supor que todas essas aplicações serão alteradas e implantadas no mesmo momento (Domingues, 2011). Por esse motivo Ambler propôs que se tenha um período de transição entre o início e o fim da refatoração, no qual o esquema antigo e o esquema novo coexistam (Domingues, 2014). Conforme ilustra a figura 2, em que um ciclo de vida de uma refatoração de banco de dados, tenha-se um período de transição entre o início e o fim de uma refatoração, em que os dois esquemas, o esquema antigo e o esquema novo existam nesse contexto.

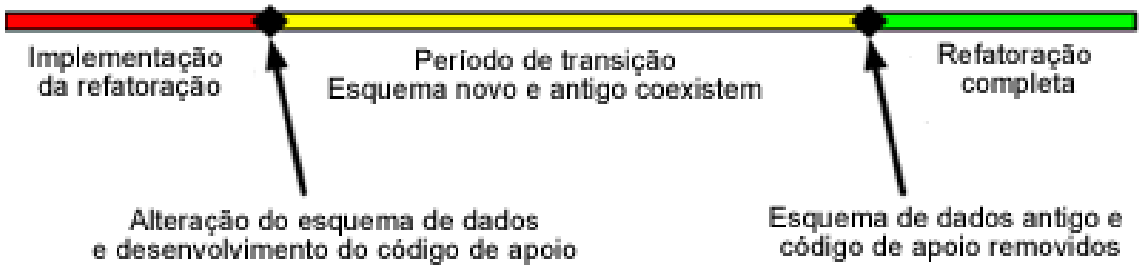


Figura 2. Ciclo de vida de uma refatoração
Fonte: adaptado de Amber; Sadalage apud Domingues (2014)

Ambler Apud Domingues (2014) destacam alguns problemas no banco de dados que podem exigir refatorações como: tabelas ou colunas com múltiplas funções; tabelas com muitas colunas, e tabelas com colunas ou associações obsoletas.

Para abranger todos esses problemas e outros que podem interferir no bom desempenho do modelo de dados, e também para atender mudanças de requisitos, Ambler e Saladage (2006) elaboraram um catálogo para as refatorações que foi amplamente discutido por Domingues (2014).

As refatorações de banco de dados são divididas em seis categorias: estrutural, qualidade de dados, integridade referencial, arquitetural, método e, transformações-não-refatoração, nas seções seguintes serão apresentadas o conceito de cada categoria de refatoração, e, as ações que cada categoria inclui em um esquema de banco de dados.

- Estrutural: As refatorações estruturais têm como objetivo melhorar o esquema do banco de dados. Mudança de requisitos são as principais causas para se alterar a estrutura do esquema do banco de dados. As refatorações estruturais podem incluir alterações no esquema do banco de dados, tais como (Domingues, 2014): Remover tabelas, colunas ou visões; Dividir ou unir tabelas ou coluna; Renomear tabelas, colunas ou visões.

- Qualidade de dados: As refatorações de qualidade de dados têm como objetivo aumentar as validações e diminuir as inconsistências do banco. As refatorações de qualidade de dados podem incluir ações como (Domingues, 2014): Introduzir restrição de coluna, valor padrão ou formato comum; Aplicar código ou tipo padrão; Consolidar estratégias chaves; Remover restrição de coluna, valor padrão ou restrição de não nulo; Alterar colunas para receber valore nulos.

- Integridade Referencial: As refatorações de integridade referencial têm como objetivo manter consistentes as referências entre tabelas, incluindo ou removendo regras, armazenando históricos ou executando lógicas de aplicação no banco de dados. As refatorações de integridade referencial podem incluir ações como (Domingues, 2014): Adicionar ou remover restrição de chave estrangeira; Adicionar ou remover a exclusão em cascata.

- Arquitetural: O objetivo principal das refatorações arquiteturais é mover uma parte do código dos aplicativos para o banco de dados (lógica do banco de dados). Os objetivos secundários são organizar, padronizar, e melhorar o desempenho do banco de dados (Domingues, 2014). As refatorações arquiteturais podem incluir ações como: Introduzir índice, tabela espelho, tabela somente para leitura; Adicionar Métodos (Stored Procedures).

- Método: As Refatorações de Método consistem em buscar por uma melhor qualidade em Stored Procedure, Function ou trigger. As Refatorações de Método estão divididas em refatorações que mudam a interface de um banco de dados por meio de programas externos e refatorações que fazem mudanças no interior de um método (Dalpra e Araújo, 2012).

A refatoração de banco de dados Método, é uma mudança que melhora a qualidade de um procedimento armazenado, função armazenada ou gatilho de modo a melhorar o seu projeto de banco de dados, sem alterar sua semântica (Agiledata, 2015).

- Transformação Não Refatoração: As refatorações de transformações são mudanças que alteram a semântica do esquema do banco de dados, adicionando novos recursos ao mesmo. As Refatorações de Transformação são as seguintes (Dalpra e Araújo, 2012): Insert Data; Introduce New Column; Introduce New Table; Introduce View; Update Data.

A transformação não refatoração é uma mudança que afeta a semântica de seu esquema de banco de dados, adicionando novos elementos ao mesmo ou modificando elementos existentes (Agiledata, 2015).

Nem todas as alterações no esquema de banco de dados podem ser definidas como refatorações. Algumas alterações inserem novos elementos ao esquema do banco de dados. A essas alterações é dado o nome de transformações (Amber; Sadalage Apud Domingues, 2014). As transformações estruturais podem incluir, no esquema do banco de dados, tabelas, colunas, visões (Domingues, 2014).

Algumas refatorações propostas por Ambler e Sadalage Apud Domingues (2014)

necessitam de replicação para manter os dados atualizados, durante o período de transição, tais como: renomear tabelas ou colunas; consolidar estratégias de chaves e dividir ou unir tabelas ou colunas. Para esses casos Ambler propõe duas estratégias que necessitam de sincronização dos dados: o uso de triggers síncronos e processo em lote (batch) (Domingues, 2014).

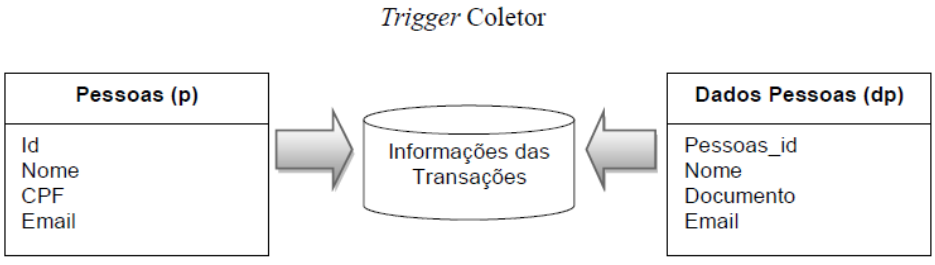
O código do trigger é acoplado a uma tabela e é disparado a cada transação executada. O banco de dados executa o código de apoio, passando para o trigger informações como qual operação foi executada, a tabela, coluna ou linha onde os dados serão atualizados. A sincronização feita em lote baseia-se em código de apoio que percorre toda a tabela origem para descobrir o que foi alterado e acrescentado, comparando a tabela destino para manter a sincronização dos dados (Domingues, 2014).

Ambler e Sadalage Apud Domingues (2014) enfatizam que o uso de trigger síncronos é a melhor solução para a replicação de dados durante a refatoração, mas segundo Domingues (2011) a utilização de triggers síncronos como código de apoio, apresenta limitações, como a necessidade de uma codificação específica para realizar a sincronização e a necessidade de tratar possíveis ciclos dos gatilhos disparados, essas dificuldades podem causar lentidão nas transações, pois elas só serão finalizadas ao término do último gatilho da sequência.

Para superar essas dificuldades, uma abordagem alternativa consiste minimizar os efeitos negativos do processo síncrono, substituindo-o por uma abordagem assíncrona para os casos de refatorações. Domingues definiu três etapas para o processo de replicação de dados durante uma refatoração: Coleta da transação, Mapeamento e Execução. Tais etapas serão descritas a seguir (Domingues, 2014).

- Etapa 1 Coleta da transação: Utiliza-se um trigger denominado “trigger coletor”, que é utilizado para todos os tipos de refatorações existentes, com o objetivo de capturar todas as informações da transação, qual foram a operação (inserção, exclusão ou atualização), quais os valores anteriores existentes na tabela e quais os novos valores usados para a atualização (Domingues, 2014). Conforme é ilustrado na figura 3 o trigger coletor.

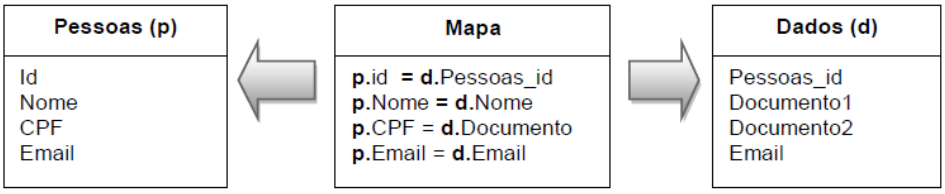
Figura 3. Replicação de dados Assíncrona: etapa de coleta de dados



Fonte: Adaptado de Domingues (2014)

- Etapa 2 Mapeamento: Nessa etapa são definidos os mapeamentos de origem e destino das transações (Domingues, 2014). Conforme é ilustrado na figura 4 o mapeamento.

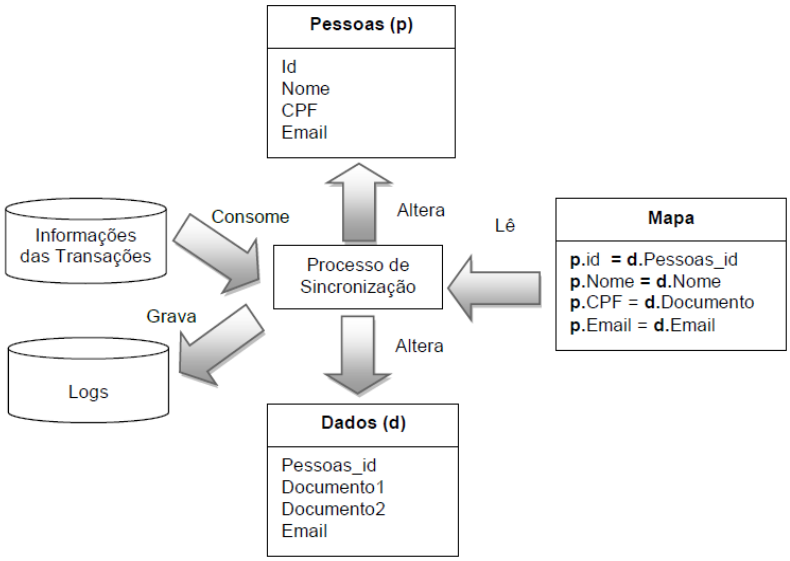
Figura 4. Replicação de dados Assíncrona: Etapa de mapeamento



Fonte: Adaptado de Domingues (2014)

- Etapa 3 Execução: Nesta etapa é disparado um processo que sincroniza o esquema antigo com o novo. Assim, as informações da coleta são processadas e são realizados os mapeamentos e gravados os logs de execução (Domingues, 2014). Conforme é ilustrada na figura 5 a execução.

Figura 5. Replicação de dados Assíncrona: Etapa de execução



Fonte: Adaptado de Domingues (2014)

Essa divisão em etapas auxilia no desenvolvimento de ferramentas automáticas para realizar as refatorações, permitindo aos desenvolvedores definir mapeamentos distintos para cada tipo de refatoração, pois apenas será codificado um *trigger* para todas as refatorações. Esta solução detecta circularidade dos *triggers* na etapa de execução, suas transações são rápidas com disparo de apenas um *trigger*, gera *log* com possíveis erros de execução, possibilita a construção de aplicativos automatizados para refatoração e não exige que as consultas SQL sejam reescritas dinamicamente para as diversas versões existentes do banco de dados (Domingues, 2014).

As abordagens síncrona e assíncrona têm como limitação não proporcionar uma organização das refatorações em conjuntos, não permitindo a realização de uma grande alteração no banco de dados. Esses conjuntos de refatorações devem ser planejados para modificar o esquema de um banco de dados já existente, sem causar perda de dados

(Domingues 2014).

Na organização de uma sequência de refatorações, deve-se pensar nas possibilidades de realizar uniões (*merges*), alinhamentos e intercalações de refatorações.

Uma organização bem feita permitirá minimizar a duração de uma grande alteração de banco de dados (Domingues, 2014).

3.1 Vantagens e Desvantagens da Refatoração

Em projetos concluídos, a refatoração pode ser vantajosa por melhorar de forma significativa um projeto mal elaborado. Ou seja, ao refatorar o banco de dados continuamente, é possível corrigir problemas graves, caso existam, que podem se tornar evidentes à medida que se aplicam as refatorações. Além desse benefício, a refatoração pode impedir que o projeto se deteriore com as possíveis alterações que tendem a ocorrer ao longo do tempo (Miguel e Araújo, 2011).

Outra vantagem que se torna evidente ao se usar as técnicas de refatoração é o aprendizado. Refatorar ajuda a tornar mais fácil a tarefa bastante comum de analisar e entender os bancos de dados. Nesse processo, é preciso investir intensamente no aprendizado do que a refatoração realmente faz e aplicar este aprendizado no banco de dados. Num nível mais alto, a refatoração permite aprender também sobre a estrutura do projeto, já que à medida que partes do banco de dados se tornam melhor entendidos, entender o projeto como um todo se torna uma tarefa muito mais fácil (Miguel e Araújo, 2011).

A velocidade é outra vantagem da refatoração, por que torna mais rápido desenvolver software por todos os motivos explicados anteriormente. Quando se tem um banco de dados e projeto mais claros e fáceis de entender, com maior nível de conhecimento pelos desenvolvedores e com menos falhas, consequentemente pode-se ter maior velocidade de desenvolvimento e manutenção a menores custos (Miguel e Araújo, 2011).

3.2 Diferenças entre a refatoração de código fonte e a refatoração de banco de dados

A refatoração consiste em melhorar a estrutura ou legibilidade de um software ou de um sistema de software sem que seu comportamento seja alterado. O processo de refatoração foi inicialmente utilizado por desenvolvedores que se utilizavam da linguagem de programação orientada a objetos Smalltalk. Posteriormente tornou-se essencial no desenvolvimento de frameworks, já que estes, em geral, não ficam totalmente prontos em uma primeira tentativa, tendo evolução no decorrer do tempo. Como o código será lido e modificado com certa frequência, a refatoração torna-se imprescindível. Usando refatoração, um projeto ruim ou caótico pode ser transformado em algo bem projetado. A junção de pequenas alterações melhora radicalmente o projeto (Dalpra e Araújo, 2012).

Já a refatoração de banco de dados propõe-se a realizar os processos de alteração do banco de maneira diferente, com ganho significativo em eficiência, em comparação à mesma aplicação no passado. É similar à refatoração de código fonte, porém aplica-se a bancos de dados. Essa técnica consiste em um método disciplinado de reestruturação de tabelas em um esquema de banco de dados, a qual ocorre em alterações gradativas. Por meio desta, busca-se o aprimoramento da estrutura de um banco relacional, otimizando consultas, e identificando pontos de melhoria junto às tabelas (Dalpra e Araújo, 2012).

4 PROCESSO DE REFATORAÇÃO EM BANCO DE DADOS

A seguir apresenta-se o processo de refatoração de banco de dados; são descritas as etapas que uma refatoração em banco de dados deve conter, levando em conta a preservação da semântica e comportamento do banco de dados em questão.

Por que refatorar o banco de dados? Há duas razões fundamentais para querer refatorar o banco de dados, são essas razões (Agiledata, 2015):

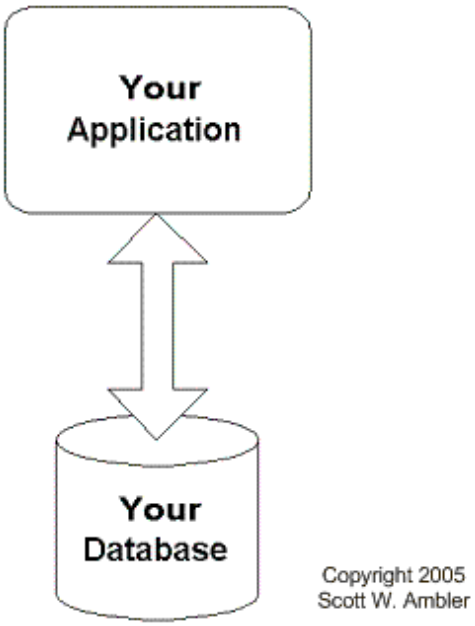
1. Para corrigir segurança em bases de dados legadas existentes. A linha inferior mostra que bancos de dados legados não irão corrigir a si mesmos, e que a partir de um ponto técnico de refatoração de banco de dados vista, é uma forma segura, simples de melhorar os dados e o banco de dados, a qualidade ao longo do tempo.
2. Para apoiar o desenvolvimento evolutivo. Processos de desenvolvimento de software modernos, como o RUP, XP, e AUP, todo o trabalho em um evolutivo se não forma ágil.

A Semântica informativa refere-se ao significado da informação no banco de dados a partir do ponto de vista dos usuários dessa informação. Para preservar a semântica informativa implica que quando você mudar os valores dos dados armazenados em uma coluna os clientes de que a informação não deve ser afetada pela melhoria. Da mesma forma, em relação à semântica comportamental o objetivo é manter a funcionalidade da caixa preta o mesmo código, fonte que trabalha com os aspectos de mudanças de seu esquema de banco de dados, que deve ser reformulado para realizar a mesma funcionalidade que antes (Agiledata, 2015).

Acoplamento, em bancos de dados relacionais é uma medida do grau de dependência entre dois itens, ao acoplar duas coisas se tem a chance de que ocorra a mudança em um que consequentemente exigirá uma mudança no outro. O acoplamento é a "raiz de todo mal", quando se trata de refatoração de banco de dados, se as coisas que estão em seu esquema de banco de dados são acopladas ao que são mais difíceis, os esquemas de banco de dados relacionais são automaticamente acopladas a uma grande variedade de coisas, que geralmente são (Agiledata, 2015): Seu código fonte da aplicação; Outro código fonte da aplicação; Código-fonte de carregamento de dados; Código-fonte extrair dados; Frameworks de persistência / camadas; Seu esquema do banco; Scripts de migração de dados; Código de ensaio; Documentação.

A figura 6 ilustra o melhor cenário para a refatoração em banco de dados, isso quando existe apenas uma aplicação interligada diretamente ao banco de dados existente.

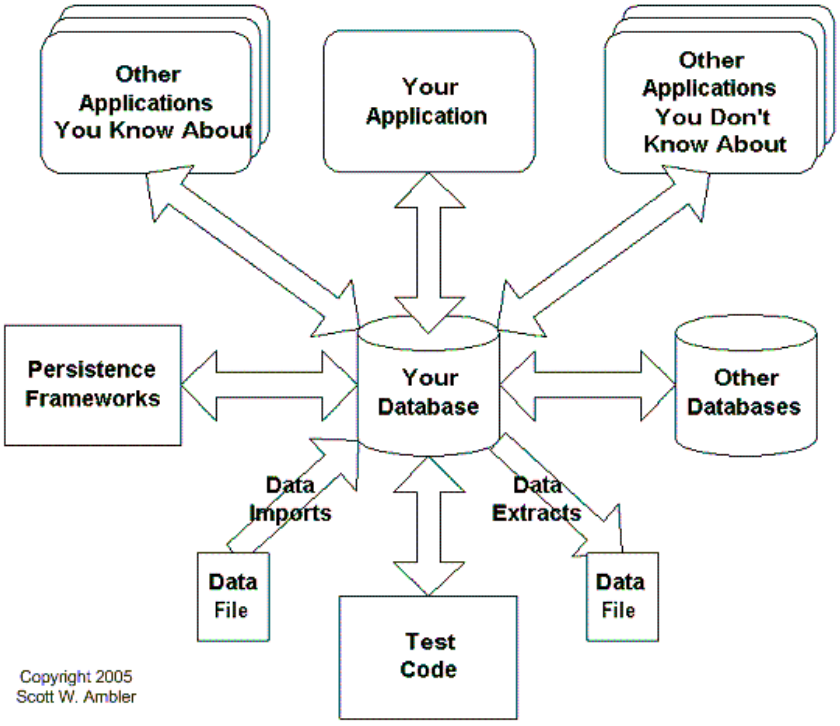
Figura 6. O melhor cenário



Fonte: http://agiledata.org/essays/databaseRefactoring.html

A figura 7 demonstra o pior cenário de refatoração em banco de dados, devido ao acoplamento.

Figura 7. O cenário de pior caso



Fonte: http://agiledata.org/essays/databaseRefactoring.html

Uma pequena transformação em seu esquema de estendê-lo, tal como a adição de uma nova coluna ou tabela, não é uma refatoração de banco de dados porque a mudança aumenta seu design. Um grande número de pequenas mudanças simultaneamente aplicadas ao seu esquema de banco de dados, como a renomeação de dez colunas, não é considerado uma refatoração de banco de dados, porque esta não é uma única e pequena mudança. Refatorações de banco de dados são pequenas alterações em seu esquema de banco de dados que melhora a sua concepção, preservando a semântica comportamental e informativa (Agiledata, 2015).

4.1 Como Refatorar o Banco de Dados

Antes de descrever os passos para refatorar um banco de dados, é preciso destacar uma questão crítica: O cenário simples destacado na figura 6 implicará a fazer coisas diferentes do que o cenário altamente acoplado da Figura 7. O processo fundamental por si, só permanece o mesmo, apesar da dificuldade de programar refatoração de banco de dados, se os acoplamentos do banco de dados aumentarem também. (Dias Neto, 2011).

Caso esteja em um cenário simples, não é necessário fazer o trabalho descrito abaixo, pode- se simplesmente refatorar o esquema de banco de dados e o código da aplicação em paralelo e implantá-los simultaneamente. Aqueles que se encontrar em um cenário mais complexo não terão este privilégio (Dias Neto, 2011).

Esta seção assume que os ambientes técnicos e culturais estão organizados para apoiar a refatoração do banco de dados. A refatoração de banco de dados pode ser vista como um processo de 3 passos (Dias Neto, 2011):

1. Começar em seu ambiente de desenvolvimento.
2. Programar em seu ambiente de integração.
3. Instalar em ambiente de produção.

Passo 1: Comece pelo ambiente de desenvolvimento

O ambiente de desenvolvimento é um ambiente técnico onde o software, incluindo o código da aplicação e esquema de banco de dados, é desenvolvido e testado (testes de unidade). A necessidade de refatorar um esquema de banco de dados é tipicamente identificada por um técnico que está tentando implantar um novo requisito ou que está corrigindo um defeito (Dias Neto, 2011).

O DBA e o desenvolvedor tipicamente irão seguir alguns ou todos os passos para implantar uma ou um conjunto de refatorações em banco de dados, que serão descritas nas seções seguintes (Dias Neto, 2011): Verificar que uma refatoração de banco de dados é necessária; Escolher a refatoração de código mais apropriada; Depreciar o esquema original; Escrever testes de unidade; Modificar o esquema do banco de dados; Migrar os dados de origem (que estavam na versão original); Atualizar os programas de acesso externo; Atualizar os scripts de migração dos dados; Executar testes de regressão; Divulgar a refatoração.

- Verificar que a refatoração do banco de dados é necessária: A primeira coisa que o DBA faz é tentar determinar se a refatoração de banco de dados de fato precisa ocorrer e se ela é a escolha correta a ser realizada. A segunda coisa é avaliar internamente a probabilidade que a mudança é realmente necessária. Isso normalmente é decidido baseando-se nas experiências anteriores do DBA e do desenvolvedor. A próxima atividade que o DBA faz é avaliar o impacto geral da refatoração (Dias Neto, 2011).

- Escolher a refatoração mais adequada ao banco de dados: Uma habilidade importante que os DBAs necessitam, é o entendimento de que você normalmente tem várias opções para a implementação de novas estruturas de dados, e nova lógica dentro de um banco de dados. (Agiledata, 2015).

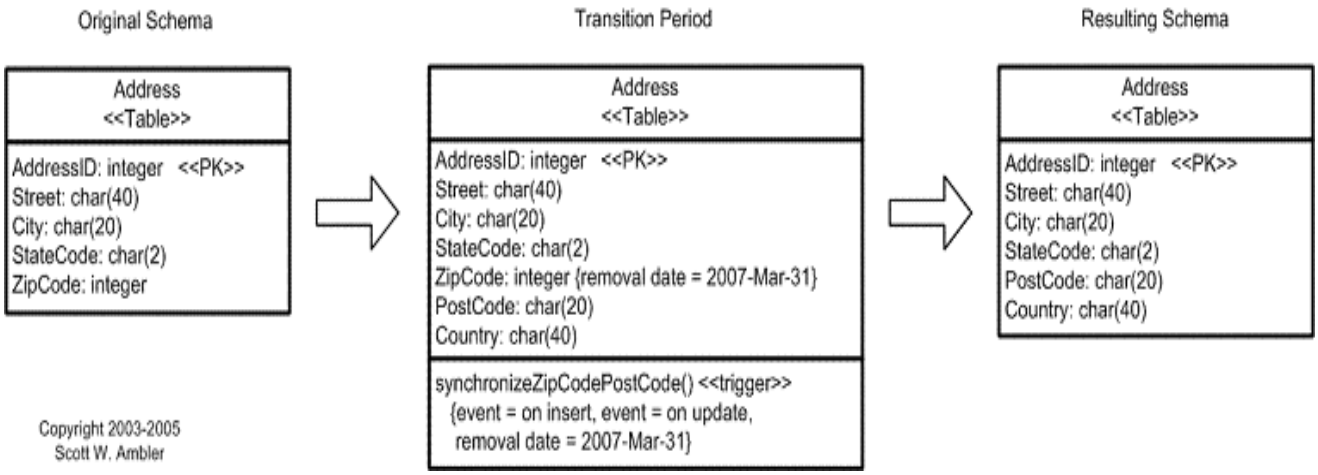
- Fazer testes de unidade: Como refatoração de código, refatoração de banco de dados, está habilitada pela existência de um conjunto de testes abrangente no qual você sabe que pode mudar seu esquema de banco de dados com segurança se você constatar que o banco de dados ainda funciona após a mudança (Agiledata, 2015).

- Depreciar o esquema original: Uma técnica efetiva de refatoração é usar um período de depreciação para a porção original do esquema que está sendo alterado. É observado que você pode simplesmente fazer a mudança em seu esquema de banco de dados instantaneamente, que em vez de precisar trabalhar com os esquemas velho e novo em paralelo por um período para prover um tempo para que equipes de desenvolvedores de outras aplicações refatorem e reimplantem seus sistemas (Dias Neto, 2011).

A Figura 8 ilustra como essa ideia funcionaria quando aplicada o Substituir da coluna zip code (CEP) por refatoração de banco de dados para post code (código postal). Observe as mudanças entre o esquema original e o esquema durante o período de transição (Agiledata, 2015).

O código postal foi adicionado como uma coluna, a coluna Zip code (CEP) foi marcada como obsoleta, porque uma data de remoção foi atribuída à mesma usando uma UML chamada variável. Um gatilho também foi introduzido para manter os valores contidos nas duas colunas sincronizados, partindo do princípio de que o novo código do aplicativo irá trabalhar com código postal, mas não deve se esperar manter a coluna Zip code (CEP) até à data, e que o código do aplicativo mais antigo que não foi reformulado para usar o novo esquema não vai saber manter código postal até à data. Esse gatilho é um exemplo de código de andaimes banco de dados, código simples e comum que é necessário para manter seu banco de dados "colados um ao outro", este código foi atribuído à mesma data de remoção como do CEP, como ilustra a figura 8 (Agiledata, 2015).

Figura 8. Refatorando tabela de endereço



Fonte: http://agiledata.org/essays/databaseRefactoring.html

- Modificar o esquema do banco: O desenvolvedor e DBA trabalham juntos para fazer as mudanças no ambiente de desenvolvimento. A estratégia é iniciar cada refatoração de forma simples, realizando a refatoração no ambiente de desenvolvimento. Para fazer isso, precisamos atualizar dois logs (assumindo que você não possua uma ferramenta de refatoração de banco de dados que faz isso automaticamente) (Dias Neto, 2011):

1. Log de mudanças no banco de dados. Este log contém os códigos fontes que programam todas as mudanças no esquema do banco de dados na ordem em que elas ocorrem ao longo do curso de um projeto. Quando estamos implementando uma refatoração de banco de dados, incluímos apenas as mudanças imediatas neste log.
2. Log de atualização. Este log contém o código fonte para mudanças futuras que serão implantadas no esquema de banco de dados que serão executadas após o período de transição da refatoração.

- Migrar os dados: Muitas refatorações requerem migração ou cópia dos dados a partir da versão antiga do esquema para o novo. O log de migração dos dados contém a linguagem de manipulação de dados (*Data Manipulation Language* – DML) para reformatar os limpar os dados de origem ao longo do projeto (Dias Neto, 2011).

- Atualizar programas externos: Os programas que acessam a parte do esquema de banco que está sendo refatorado devem ser atualizados para trabalhar com a nova versão do esquema do banco de dados. Todos esses programas devem ser reformulados e em seguida, implantados em produção antes do término do período de transição (Agiledata, 2015).

- Executar testes de regressão: Uma vez que as mudanças no código da aplicação e no esquema do banco de dados foram realizadas, precisa-se então executar os testes de regressão. Como testes realizados com sucesso descobrem problemas, precisa-se reavaliar os itens alterados até que estes estejam corretos. Uma vantagem significativa das refatorações de banco de dados serem pequenas mudanças é que se os testes falham, é mais simples de identificar a origem do problema. Quanto maior for a mudança, mais difícil será

identificar o problema que originará um eventual problema, e portanto o esforço aumentará. Isso confirma que desenvolvimento em passos pequenos e incrementais funciona muito bem na prática (Dias Neto, 2011).

- Divulgar as mudanças que fez: Como seu banco de dados é um recurso compartilhado (minimamente ele é compartilhado pela equipe de desenvolvimento da aplicação, isso se não for por várias outras aplicações), o DBA precisa comunicar às mudanças que foram feitas. Se ainda não foi feito, então devemos atualizar o modelo de dados físico do banco de dados. Uma alternativa interessante é usar ferramentas que a partir do modelo físico geram os scripts para alteração do banco de dados (Dias Neto, 2011).

- Controlar a Versão de trabalho: A habilidade crítica para desenvolvedores ágeis é o hábito de colocar todo o seu trabalho sobre gerenciamento de configurações de controle, verificando-o em uma ferramenta de controle de versão. No caso do banco de dados de refatoração isso inclui qualquer DDL (Data Definition Language) que você criou scripts de alteração, scripts de migração de dados, dados de teste, casos de teste, teste de código de geração de dados, documentação e modelos. Isto é, além de os artefatos centrados em aplicações que normalmente versão tratar o seu artefato orientado para o banco de dados da mesma maneira que você iria tratar outros artefatos de desenvolvimento você deve estar ok (Agiledata, 2015).

Passo 2: Implementação do ambiente de integração

Após vários dias de trabalho estaremos prontos para implantar a refatoração de banco de dados no ambiente de integração. A razão pela qual precisamos aguardar para fazer isso é que precisamos dar tempo às equipes de outros sistemas refatorarem seus códigos para usarem o novo esquema de banco de dados (Dias Neto, 2011).

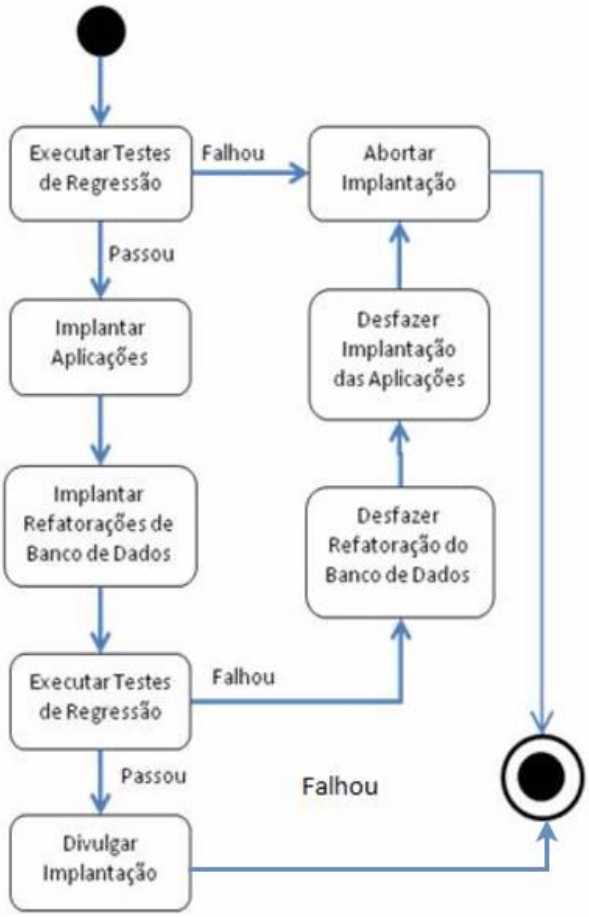
As equipes que escolherem encapsular o acesso ao banco de dados usando um framework de persistência julgarão simples o tratamento às alterações no esquema de banco de dados e, portanto descobrirão que poderão encurtar o período entre a implementação da refatoração do banco de dados no ambiente de desenvolvimento e em seu ambiente de integração do projeto. Isso ocorre pelo fato do esquema do banco de dados ser representado em metadados, portanto muitas mudanças no esquema do banco de dados irão requerer ajustes apenas nos metadados e não no código fonte atual (Dias Neto, 2011).

Passo 3: Instalação em produção

Instalar em produção é a parte mais difícil do processo de refatoração de banco de dados, mais ainda no cenário complexo da Figura 7. Geralmente não implantaremos refatorações de banco de dados apenas para o nosso sistema, em vez disso iremos implantá-las como parte do desenvolvimento geral de um ou vários sistemas (Dias Neto, 2011).

A implantação é mais fácil quando temos apenas uma aplicação e um banco de dados para atualizar, e este cenário ocorre na prática, mas de fato precisamos considerar o cenário onde iremos implantar vários sistemas e várias fontes de dados ao mesmo tempo. A Figura 9 apresenta uma visão geral dos passos para implantação da refatoração em um ambiente de produção (Dias Neto, 2011).

Figura 9. Os passos para implantação de refatoração de banco de dados



Fonte: http://www.devmedia.com.br/o-processo-de-refatoracao-de-banco-de-dados-artigo-revista-sql-magazine-86/20400

5 ESTUDO COMPARATIVO DE TÉCNICAS DE REFATORAÇÃO DE BANCO DE DADOS DESCRITOS NA LITERATURA

Nessa seção serão contextualizadas tabelas com as seis categorias de refatoração de banco de dados, e suas técnicas de refatoração, ao todo são cerca de 70 técnicas de refatoração de banco de dados presente nas tabelas a seguir.

A tabela 1 demonstra a categoria de refatoração estrutural, com 17 técnicas de refatoração, às mesmas discutidas no livro e no catálogo web de refatorações de Scott W. Ambler. Serão mostradas também, as técnicas de refatoração abordadas pelos autores nos trabalhos correlatos, todas marcadas com x, e o total de técnicas de refatoração que cada autor abordou.

Tabela 1. Refatorações Estruturais

Técnicas de Refatoração		Autor 1	Autor 2	Autor 3
Categoria 1	Refatoração	Scott W. Ambler Autor do Livro de Refatoração em Banco de Dados	Márcia Beatriz Pereira Domingues	Helves Humberto Domingues
Estrutural	Eliminar Coluna	X	X	X
Estrutural	Eliminar Tabela	X		X
Estrutural	Eliminar Visão	X		X
Estrutural	Introduzir Coluna Calculada	X		X
Estrutural	Introduzir chave de identificação	X	X	X
Estrutural	Unir Colunas	X	X	X
Estrutural	Unir Tabelas	X		X
Estrutural	Mover Coluna	X		X
Estrutural	Renomear Coluna	X		X
Estrutural	Renomear Tabela	X		X
Estrutural	Renomear Visão	X		X
Estrutural	Trocar Coluna	X		X
Estrutural	Trocar Coluna Complexa por Tabela	X		X
Estrutural	Trocar um para muitos por tabela associativa	X		X
Estrutural	Trocar chave de identificação por chave natural	X		X
Estrutural	Dividir Coluna	X		X
Estrutural	Dividir Tabela	X		X
Total de Refatorações = 17		Total = 17	Total = 3	Total = 17

Fonte: Autoria própria

A tabela 2 demonstra a categoria de refatoração qualidade de dados, com 13 técnicas de refatoração, às mesmas contextualizadas no livro e no catálogo web de Scott W. Ambler. Também estão evidenciadas na tabela, as técnicas de refatoração que foram abordadas pelos autores nos trabalhos correlatos, e, o total de técnicas de refatorações que cada autor abordou.

Tabela 2. Técnicas de Refatoração Qualidade de Dados

Técnicas de Refatoração		Autor 1	Autor 2	Autor 3
Categoria 2	Refatoração	Scott W. Ambler Autor do Livro de Refatoração em Banco de Dados	Márcia Beatriz Pereira Domingues	Helves Humberto Domingues
Qualidade de dados	Adicionar Tabela Descritiva	X	Não Utilizou	X
Qualidade de dados	Aplicar Código Padrão	X	Não Utilizou	X
Qualidade de dados	Aplicar Tipo Padrão	X	Não Utilizou	X
Qualidade de dados	Consolidar Estratégias de Chaves	X	Não Utilizou	X
Qualidade de dados	Remover Restrição de Coluna	X	Não Utilizou	X
Qualidade de dados	Remover Valor Padrão	X	Não Utilizou	X
Qualidade de dados	Remover Restrição de não Nulo	X	Não Utilizou	X
Qualidade de dados	Introduzir Coluna de Restrição	X	Não Utilizou	X
Qualidade de dados	Introduzir formato Comum	X	Não Utilizou	X
Qualidade de dados	Introduzir Valor Padrão	X	Não Utilizou	X
Qualidade de dados	Alterar Coluna para não Nula	X	Não Utilizou	X
Qualidade de dados	Mover Dados	X	Não Utilizou	X
Qualidade de dados	Trocar Código de Tipo por Propriedades Sinalizadoras	X	Não Utilizou	X
Total de Refatorações = 13		TOTAL = 13	Total = 0	TOTAL = 13

Fonte: Autoria própria

A tabela 3 demonstra a categoria de refatoração integridade referencial, com 7 técnicas de refatoração, as mesmas abordadas no livro e no catálogo web de Scott W. Ambler. Outra amostragem são as técnicas de refatoração que foram abordadas pelos autores nos trabalhos correlatos, como demonstrado na tabela 3 marcada com x, e o total de técnicas de refatoração que cada autor abordou.

Tabela 3. Refatorações de Integridade Referencial

Técnicas de Refatoração		Autor 1	Autor 2	Autor 3
Categoria 3	Refatoração	Scott W. Ambler Autor do Livro de Refatoração em Banco de Dados	Márcia Beatriz Pereira Domingues	Helves Humberto Domingues
Integridade Referencial	Adicionar Restrição de Chave Estrangeira	X	X	X
Integridade Referencial	Adicionar Trigger para Coluna Calculada	X		X
Integridade Referencial	Remover restrição de chave estrangeira	X		X
Integridade Referencial	Introduzir exclusão em Cascata	X		X
Integridade Referencial	Introduzir exclusão Lógica	X		X
Integridade Referencial	Introduzir exclusão Física	X		X
Integridade Referencial	Adicionar Trigger para Histórico	X		X
Total de Refatorações = 7		Total = 7	Total = 1	Total = 7

Fonte: Autoria própria

A tabela 4 demonstra a categoria de refatoração arquitetural, com 12 técnicas de refatoração, contextualizadas no livro e no catálogo web de refatorações de Scott W. Ambler. Também as técnicas de refatoração que foram elucidadas nos trabalhos correlatos desta categoria de refatoração, e o total de técnicas de refatoração abordadas por cada autor.

Tabela 4. Refatorações Arquiteturais

Técnicas de Refatoração		Autor 1	Autor 2	Autor 3
Categoria 4	Refatoração	Scott W. Ambler Autor do Livro de Refatoração em Banco de Dados	Márcia Beatriz Pereira Domingues	Helves Humberto Domingues
Arquitetural	Adicionar Métodos Crud	X		X
Arquitetural	Introduzir Tabela Espelho	X		X
Arquitetural	Adicionar Método de Leitura	X		X
Arquitetural	Encapsular tabela com uma visão	X		X
Arquitetural	Introduzir Método para Cálculo	X		X
Arquitetural	Introduzir Índice	X	X	X
Arquitetural	Introduzir tabela somente de leitura	X		X
Arquitetural	Migrar método para banco de dados	X		X
Arquitetural	Migrar método de banco de dados	X		X
Arquitetural	Trocar método por Visão	X		X
Arquitetural	Trocar visão por Método	X		X
Arquitetural	Usar Fonte de dados Oficial	X		X
Total de Refatorações = 12		Total = 12	Total = 1	Total = 12

Fonte: Autoria própria

A tabela 5 demonstra a categoria de refatoração de método, com 16 técnicas de refatoração, contextualizadas no livro e no catálogo web de refatorações de Scott W. Ambler. Nessa categoria apenas o autor do livro e do catálogo web de refatorações Scott W. Ambler abordou o assunto, e suas respectivas técnicas de refatoração de banco de dados.

Tabela 5. Refatorações de Método

Técnicas de Refatoração	Autor 1	Autor 2	Autor 3

Categoria 5	Refatoração	Scott W. Ambler Autor do Livro de Refatoração em Banco de Dados	Márcia Beatriz Pereira Domingues	Helves Humberto Domingues
Método	Adicionar parâmetro	X	Não é Citado	Não é Citado
Método	Consolidar Expressão Condicional	X	Não é Citado	Não é Citado
Método	Decompor Condicional	X	Não é Citado	Não é Citado
Método	Extrair Método	X	Não é Citado	Não é Citado
Método	Introduzir Variável	X	Não é Citado	Não é Citado
Método	Parametrizar Métodos	X	Não é Citado	Não é Citado
Método	Remover Sinalização de Controle	X	Não é Citado	Não é Citado
Método	Remoção Intermediária	X	Não é Citado	Não é Citado
Método	Remover Parâmetro	X	Não é Citado	Não é Citado
Método	Renomear Método	X	Não é Citado	Não é Citado
Método	Reordenar Parâmetros	X	Não é Citado	Não é Citado
Método	Substituir Literal Com tabela de pesquisa	X	Não é Citado	Não é Citado
Método	Substituir Expressão Aninhada Com Cláusulas Guarda	X	Não é Citado	Não é Citado
Método	Substituir parâmetro com métodos específicos	X	Não é Citado	Não é Citado
Método	Dividir Variável Temporária	X	Não é Citado	Não é Citado
Método	Algoritmo substituto	X Visualização web Indisponível	Não é Citado	Não é Citado
Total de Refatorações = 16		Total = 16	Total = 0	Total = 0

Fonte: Autoria própria

A tabela 6 demonstra a categoria de refatoração Transformações – não refatoração, com 5 técnicas de refatoração, todas contextualizadas no livro e no catálogo web de Scott W. Ambler. Nessa categoria apenas o autor do livro e do catálogo web de

refatorações as abordaram e mostraram suas respectivas técnicas de refatoração.

Tabela 6. Refatorações Transformações-Não Refatorações

Técnicas de Refatoração		Autor 1	Autor 2	Autor 3
Categoria 6	Refatoração	Scott W. Ambler Autor do Livro de Refatoração em Banco de Dados	Márcia Beatriz Pereira Domingues	Helves Humberto Domingues
Transformações – Não Refatoração	Inserir dados	X	Não é Citado	Não é Citado
Transformações – Não Refatoração	Introduzir nova coluna	X	Não é Citado	Não é Citado
Transformações – Não Refatoração	Introduzir Nova Tabela	X	Não é Citado	Não é Citado
Transformações – Não Refatoração	Introduzir Vista	X	Não é Citado	Não é Citado
Transformações – Não Refatoração	Atualizar Dados	X	Não é Citado	Não é Citado
Total de Refatorações = 5		Total = 5	Total = 0	Total = 0

Fonte: Autoria própria

Nas tabelas em questão estão marcadas com x as técnicas de refatoração de banco de dados que cada autor utilizou em seu trabalho, seja um estudo caso, como também em um caso de uso (exemplo), Ambler utilizou todas as mesmas em seu livro Refactoring Databases: Evolutionary Database Design juntamente com Sadalage, e que o mesmo também se encontra disponível no endereço eletrônico http://agiledata.org/essays/databaseRefactoring.html, na tese Domingues (2011) foram utilizadas 49 técnicas de refatoração de banco de dados de quatro categorias como estão contextualizadas nas tabelas 1, 2, 3 e, 4, e demarcadas com x em casos de uso (exemplos), já Domingues (2014) utilizou em sua tese 5 técnicas de refatoração de banco de dados de três categorias como demostra as tabelas 1, 3 e, 4 demarcadas com x as técnicas de refatoração que foram utilizadas em seu estudo de caso.

CONCLUSÃO

Este estudo apresentou técnicas de refatoração em banco de dados relacional que visam corrigir erros e anomalias que surgem com o passar do tempo, devido a imperfeições e falhas nos esquemas conceituais do banco de dados, fruto da ineficiência do processo de modelagem de dados que acarretam duplicações de informações, inconsistências, entre

outros problemas, deixando o banco de dados menos confiável e suas informações cada vez mais difíceis de serem utilizadas no processos de tomada de decisão.

A refatoração de banco de dados é composta por seis categorias que contêm 70 técnicas de refatoração, todas apontadas por Scott W. Ambler e Pramond J. Sadalage (2006), que no presente trabalho foram utilizadas seis técnicas de refatoração que são: eliminar coluna, eliminar tabela, renomear tabela, que pertencem á categoria estrutural, introduzir valor comum que pertence á categoria qualidade de dados, adicionar restrição de chave estrangeira que pertence á categoria integridade referencial, e, introduzir índice que pertence á categoria arquitetural, todas referenciadas nos trabalhos correlatos.

Foram apresentados os conceitos de refatoração de banco de dados descritas nos trabalhos correlatos, e também referenciados por outros autores que discutem o assunto em questão, no qual, apresentam as vantagens da refatoração e as diferenças entre refatorar código fonte e esquema de banco de dados. Foi descrito o processo de refatoração de banco de dados e foram relatadas as técnicas de refatoração de banco de dados em suas respectivas categorias, os autores que as utilizaram em seus trabalhos correlatos, e também a quantidade de técnicas que cada autor utilizou em seu trabalho, sendo em estudo de caso, ou casos de uso (exemplos). As técnicas de refatoração em banco de dados são divididas em 6 categorias que são: refatoração estrutural, refatoração de qualidade de dados, refatoração de integridade referencial, refatoração arquitetural, refatoração de método, e transformações não refatoração, todas essas técnicas são descritas por Ambler e Saladage (2006).

Em conjunto com as técnicas de refatoração do banco de dados, a engenharia reversa também colabora na minimização de erros nos esquemas conceituais.

O processo de refatoração em banco de dados difere-se da refatoração de código, além de manter a estrutura interna, deve-se também manter a integridade das informações e as funcionalidades do banco, caso contrário, pode haver o impedimento na refatoração. Outra variável é o acoplamento que é a medida de dependência entre o banco de dados e os sistemas legados, isto é, quanto mais alto o acoplamento mais difícil se torna a refatoração.

Para os banco de dados que apresentam anomalias informacionais causadas por inconsistências de dados, foi proposta neste estudo a aplicação das técnicas de refatoração de banco de dados das categorias estrutural, qualidade de dados, integridade referencial e arquitetural em casos de uso, que tende a minimizar erros e facilitar o entendimento das informações contidas no banco de dados, assim facilitar consultas, manter a semântica e melhorar sua estrutura, conservando suas funcionalidades originais sem adicionar novas funcionalidades, melhorando os esquemas para projetos futuros em banco de dados. O estudo comparativo das técnicas de refatoração de banco de dados descritas na literatura apresentado neste trabalho pode ajudar os projetistas a encontrarem as melhores técnicas de acordo com os problemas encontrados nos banco de dados.

REFERÊNCIAS

Catálogo de Refatorações de Banco de Dados, Disponível em: < http://www.agiledata.org/essays/databaseRefactoringCatalog.html>. Acesso em: 14 de Ago de 2015.

Chen, P. (1990) *Modelagem de dados*: A abordagem entidade-relacionamento para projeto lógico.

Dias Neto, A. C. (2011) *O processo de refatoração de banco de da*dos, Disponível em: < http://www.devmedia.com.br/o-processo-de-refatoracao-de-banco-de-dados-artigo-revista-sql-magazine-86/20400>. Acesso em: 13 de Set de 2015.

Dalpra, H. L. O; Araújo, M. A. P. (2012) *Refatoração aplicadas a bancos de dados*, Disponível em: < http://www.devmedia.com.br/refatoracao-aplicadas-a-bancos-de-dados-revista-engenharia-de-software-magazine-46/23815>. Acesso 10 em: Set de 2015.

Domingues, H. H. *Replicação assíncrona em bancos de dados evolutivos.* 2011. 150 f. Tese (Doutorado) Instituto de Matemática e Estatística da Universidade de São Paulo, 2011.

Domingues, M.B.P. *Um novo processo para refatoração de banco de dados.* 2014. 115 f. Tese (Doutorado) Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais, 2014.

Fowler, M; et. al. (1999) Refatoração: *Aperfeiçoando o projeto de código existente.*

Introdução à Normalização de dados: um banco de dados "Melhores Práticas", Disponível em: < http://agiledata.org/essays/dataNormalization.html>. Acesso em: 25de Out de 2015.

Macedo, D. (2011) *Modelagem Conceitual, Lógica e Física de Dados*, Disponível em: < http://www.diegomacedo.com.br/modelagem-conceitual-logica-e-fisica-de-dados/>. Acesso em: 10 de Out de 2015.

Miguel, M. A; Araújo, M. A. P. (2011) *Refatoração em Banco de Dados*, Disponível em: < http://www.devmedia.com.br/refatoracao-em-banco-de-dados-sql-magazine-84/19065>. Acesso em: 20de Set de 2015.

O processo de refatoração de banco de dados: Estratégias para Melhorar a Qualidade de Banco de Dados, Disponível em: < http://agiledata.org/essays/databaseRefactoring.html>. Acesso em 12 de Set de 2015.

Yong, C. S. (1990) *Banco de Dados*: Organização sistemas e administração.