colloquium exactarum

DESENVOLVIMENTO DE UMA FERRAMENTA PARA MIGRAÇÃO DE DADOS ENTRE BANCOS DE DADOS RELACIONAIS

DEVELOPMENT OF A TOOL FOR DATA MIGRATION FROM RELATIONAL DATABASES

Tamiris Fernanda Malacrida[1], Aglaê Pereira Zaupa[2], Mário Augusto Pazoti[2]

[1]Discente da Faculdade de Informática da UNOESTE. [2]Docente da Faculdade de Informática da UNOESTE.

RESUMO - Com o aumento exponencial no uso de computadores nos últimos tempos, surgem em elevado número diferentes tecnologias de gerenciamento de dados. Independente do tipo de armazenamento de dados existente, o que na última década era a melhor solução nos dias atuais pode não ser mais viável por questões financeiras, de robustez ou de compatibilidade com a tecnologia atual. Nesse sentido, surge em grande escala a necessidade de migração dos dados entre bancos de dados. Portanto, esse trabalho visa o desenvolvimento de uma ferramenta que dará suporte ao usuário durante a mudança de uma base de dados relacional para outra, bem como no processo de reengenharia dos dados, com vistas em manter os dados íntegros. A ferramenta tem como propósito principal possibilitar que tabelas, atributos e relacionamentos que compõem o antigo banco de dados sejam mapeados de acordo com os elementos da nova modelagem, contando que os dados sejam transportados para o novo SGBD – Sistema Gerenciador de Banco de Dados de forma semiautomatizada, sempre preservando a integridade da base de dados.
Palavras-chave: banco de dados; migração de dados; modelo de dados relacional.

ABSTRACT - With the exponential rise on the use of computers recently, different technologies on data management have appeared as well. Apart from the existent data storage type, what used to be the best solution, nowadays, can not be viable anymore due to financial, strength questions or compatibility with the new technologies. In this case, arises the need, in a big scale, of data migration between databases. Therefore, this work aims to the development of a tool that will support the user during the exchange from a relational database to another, as well as the data reengineering, aiming in keep the data straightforward. The tool main purpose is to enable the tables, attributes and relationships that composes the old database be mapped according to the elements in the new modeling, taking in account that the data be transported to the new database management system in a semi automated way, always preserving the integrity of the data base.
Keywords: database; data migration; relational data models.

Recebido em: 15/09/2013
Revisado em: 18/10/2013
Aprovado em: 25/11/2013

1 INTRODUÇÃO

Atualmente com a expansão da tecnologia, surgem diferentes técnicas de gerenciamento de dados, essas que vão de arquivos texto até sistemas de banco de dados relacionais de grande porte. Com isso o que na última década era a melhor solução, hoje se encontra totalmente desatualizado.

De modo que a tecnologia desatualizada não é o único motivo para que uma empresa decida realizar o processo de migração de dados. Outros motivos incluem custos dos SGBD's, melhor desempenho, consolidação de diferentes tecnologias e quantidade de recursos disponíveis nos SGBD's mais atuais.

Ao utilizar uma ferramenta que realiza a migração de dados entre bases, há uma redução significativa da necessidade de digitação das informações, pois a tarefa de digitar dados manualmente torna-se inviável dependendo do volume de dados armazenados. Primeiramente pelo tempo gasto, mas principalmente pelo comprometimento da integridade dos dados armazenados.

A motivação para o desenvolvimento dessa ferramenta surgiu devido à necessidade de realizar a migração entre bases de dados do mesmo contexto, mesmo que tenham estruturas diferentes. Existem na literatura trabalhos que apresentam alternativas para migração de dados, como as ferramentas spoon e jExodus (NETO e PASSOS, 2009). No entanto, suas propostas englobam a migração apenas de bases de dados idênticas.

O cenário típico para a migração é o processo de movimentação de dados de um repositório ou fonte de dados para outra, através de scripts ou programas automatizados, realizando a extração dos dados da antiga base de dados e exportando para a nova base de dados definida pelo usuário (BARRETTO, 2008).

Este artigo está organizado em cinco seções, onde a segunda seção apresenta o conceito de banco de dados relacionais; a terceira seção discorre sobre a migração de dados; a quarta refere-se a ferramenta desenvolvida detalhando seus principais componentes; e, finalmente a última seção aborda as conclusões e sugestões para trabalhos futuros.

2 BANCO DE DADOS RELACIONAL

Em estudos realizados por Silberschatz, Korth e Sudarshan (1999, p.84),

> Um banco de dados relacional consiste em uma coleção de tabelas, cada qual com nome único atribuído. Uma linha em uma tabela representa uma relação entre um conjunto de valores. Uma vez que essa tabela é uma coleção desses relacionamentos, existe uma íntima correspondência entre o conceito de tabela e o conceito matemático de relação, do qual o modelo de dados relacional extrai seu nome.

A composição do modelo relacional é organizada em tabelas, atributos, chaves e relacionamentos, sendo que uma tabela é definida como um conjunto não ordenado de linhas, cada linha é composta por uma série de atributos. Os atributos são elementos de dados que descrevem o valor de uma propriedade particular em uma tabela. As chaves correspondem ao conceito básico que estabelece as relações entre linhas de tabelas de um banco de dados relacional, dentro do conceito de chaves existem três tipos a serem mencionados (HEUSER, 2001): i) chave primária que é denominada como o atributo identificador de cada tabela; ii) chave estrangeira, responsável por gerar os relacionamentos dos registros entre uma tabela e outra; iii) chave alternativa caracterizada como um atributo de valor único na tabela, embora ele seja único, pode não ter valor associado.

Pela composição do banco de dados relacional, por último há o relacionamento, que é considerado como uma associação de uma ou várias tabelas do banco de dados e composto pelas cardinalidades:

- **Um para Um (1:1):** Um elemento da entidade A está associado no máximo a uma ocorrência da entidade B, e uma entidade B está associada à no máximo uma ocorrência da entidade A.
- **Um para Muitos (1:N):** Um elemento da entidade A está associado a várias ocorrências da entidade B. Um elemento da entidadeB, entretanto deve estar associadoa no máximo uma ocorrência da entidadeA.
- **Muitos para Um (N:1):** Umelemento da entidade A está associada à no máximo uma ocorrência da entidadeB. Uma entidade B, entretanto, pode estar associada a um número qualquer de ocorrências na entidade A.
- **Muitos para Muitos (N:N):** Um elemento da entidade A está associada a qualquer número de ocorrências da entidade B e uma entidade B está associada a um número qualquer de ocorrências da entidades A.

3 MIGRAÇÃO

Em banco de dados, migração é o processo de transferência de dados de uma determinada base de dados origem, para outra base de dados definida como destino.

A transferência dos dados entre essas fontes é feita através do processo conhecido como ETL – Extração, Transformação e Carga.

3.1 ETL

ETL (Extract, Transform and Load) é um processo que se destina à extração, transformação e carga de dados (PRIMACK,

2008). Segue a representação das etapas do processo ETL na Figura 1, considerando a proposta desse trabalho em que há a extração dos dados na base de origem (SQL Server), sua transformação e posteriormente a carga desses dados no SGBD de destino (Oracle).

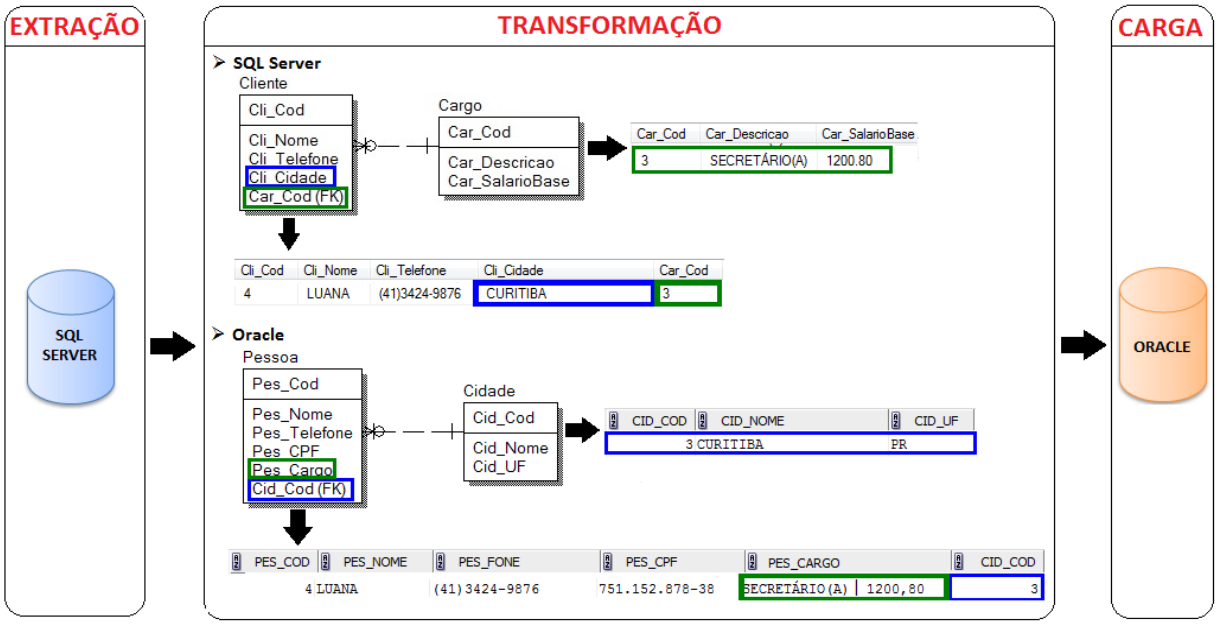


Figura 1. Processo de ETL através de bases de dados semelhantes

3.1.1 EXTRACT

A palavra *Extract* refere-se à extração. Ao iniciar o processo de ETL, existe a extração dos dados, que é proveniente da base de dados definida pelo usuário como base de origem. Nesta fase é necessário identificar forma de armazenamento, estrutura e modelagem dos dados a serem extraídos, além da necessidade de viabilizar através da ferramenta de extração um meio de acesso a estes dados de origem. Como apresentado na Figura 1, o processo de extração dos dados é realizado no banco de dados SQL Server.

3.1.2 TRANSFORM

A palavra *Transform* refere-se à transformação, em posse dos dados, são elaboradas todas as regras, que consiste em vincular os campos do sistema de origem para o sistema de destino. Durante essa fase é realizada a validação e/ou transformação evitando qualquer inconsistência nas informações a serem migradas, para que os dados a serem inseridos estejam em conformidade com a base receptora, trabalhando da melhor forma, para não perder informações no caso de bases que

não sejam idênticas, porém do mesmo contexto podendo ter estruturas diferentes.

Nesse processo é realizada a verificação de compatibilidade de tipos de dados, tamanho dos dados, atributos obrigatórios/opcionais, além da migração de atributos com estruturas diferentes entre si, porém com equivalências, respeitando todas as pré-definições e críticas apontadas na programação. Conforme apresentado na Figura 1, o processo de transformação é realizado de forma que, considerando a tabela Cliente com o atributo Cli_Cidade na base de dados origem (SQL Server), sendo mapeado para o atributo Cid_Cod da tabela Pessoa da base de dados destino (Oracle), onde o atributo do destino é uma chave estrangeira. Portanto é escolhido na tabela de referência do Cid_Cod(FK)que éa tabela Cidade, qual o atributo dessa tabela tem valor correspondente ao valor que está no atributo da origem Cli_Cidade. Em seguida, é retornado para inserção no atributo Cid_Cod(FK)da tabela Pessoa, o valor do atributo Cid_Cod(PK)que referencia o registro na tabela Cidade, em que o atributo Cid_Nome tem como valor o mesmo resultado do atributo Cli_Cidade da tabela Cliente da origem. Essa é uma das formas de transformação apresentada pela ferramenta.

3.1.3 LOAD

A palavra *Load* refere-se à carga, o processo de carga é responsável por gravar os dados extraídos, tratados e transformados de forma íntegra e consistente no banco de dados de destino, tratando conflitos existentes de chaves primárias e atualizações de chaves estrangeiras de acordo com seus respectivos valores na base de dados de destino definida pelo usuário. Como apresenta a figura 1, a carga dos dados está sendo realizada na base de destino Oracle.

4 FERRAMENTA

O processo de migração é realizado quando existem duas bases de dados definidas, sendo uma delas a origem e a outra o destino. Portanto, esse processo de migração deve ser realizado pela ferramenta desenvolvida neste trabalho.

A ferramenta foi implementada na linguagem C#, com ambiente de desenvolvimento Microsoft Visual Studio Ultimate 2010.

Foram considerados dois SGBD's – Sistemas Gerenciadores de Banco de Dados, são eles: SQL Server como base de dados de origem e Oracle como base de dados de destino. Essa limitação existe em decorrência da diversidade de características específicas dos SGBD's.

As bases de dados apresentadas podem ser bases idênticasou base de dados do mesmo contexto podendo ter estruturas diferentes.

4.1 ESTRUTURA DOS METADADOS

Uma característica fundamental da abordagem de um banco de dados é que, os SGBD's não possuem somente a estrutura de armazenamento dos dados, mas também uma completa definição e/ou descrição da estrutura desse banco e de suas restrições. Essa definição está armazenada no catálogo do SGBD, o qual contém informações como a estrutura de cada tabela, o tipo e o formato de armazenamento de cada item de dado e várias restrições sobre eles, porém essas informações armazenadas no catálogo são conhecidas também como metadados. "Metadados são os itens particulares que descrevem a estrutura de um banco de dados, muitas vezes são ditos como dados que descrevem os dados." (MONTEIRO; LIFSCHITZ; BRAYNER, 2007).

Para o processo de desenvolvimento da ferramenta foi necessário o acesso aos metadados dos bancos de dados SQL Server e Oracle.Como pode-se observar, a figura 2 ilustra uma parte da estrutura dos metadados do SGBD de origem, o SQL Sever.

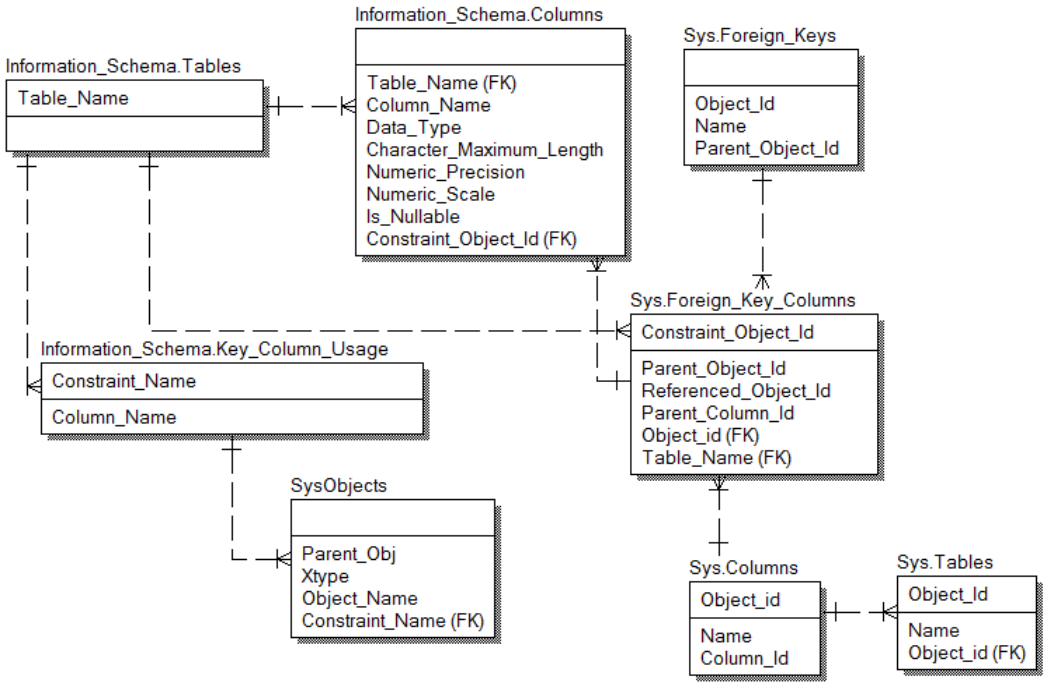


Figura 2. Estrutura dos Metadados do SQL Server

A figura 3 ilustra uma parte da estrutura dos metadados do SGBD definido como destino, o Oracle.

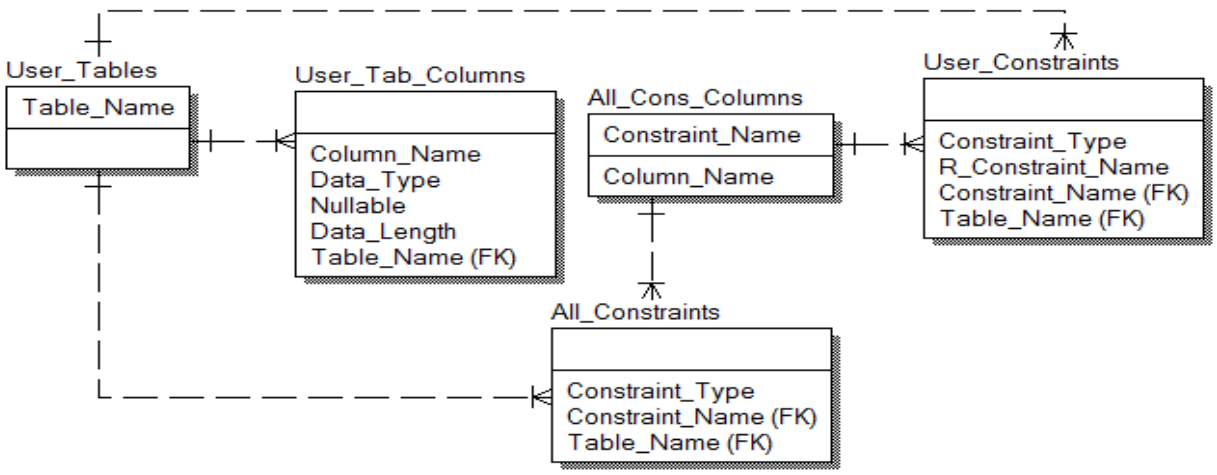


Figura 3. Estrutura dos Metadados do Oracle

Vale ressaltar que apenas as partes relevantes e utilizadas de cada metadados dos SGBD's foram apresentadas, não necessitando de toda a estrutura existente.

4.2 ESTUDO DE CASO

Para apresentação da ferramenta, foi produzido um estudo de caso que abrange todas as possibilidades de mapeamentos que a ferramenta permite, essa modelagem trata-se de um sistema de controle vendas.

Para exemplificar a situação, as figuras 4 e 5 os modelos de dados de origem e destino, representando os casos que podem ocorrer em uma migração de bases de mesmo contexto com estruturas diferentes.

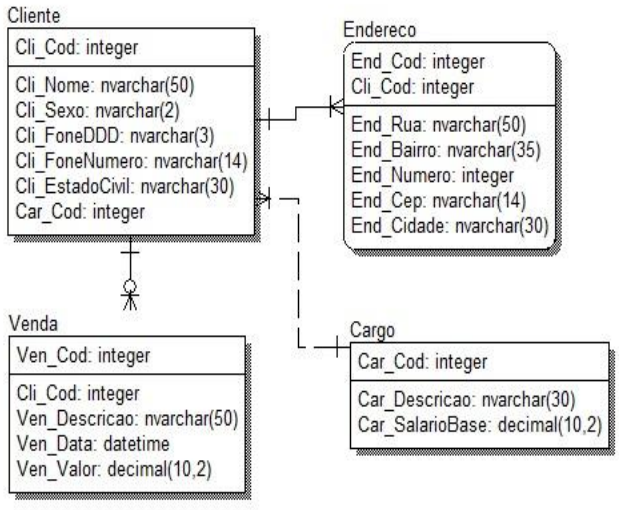


Figura 4. Base de Dados de Origem –SQL Server

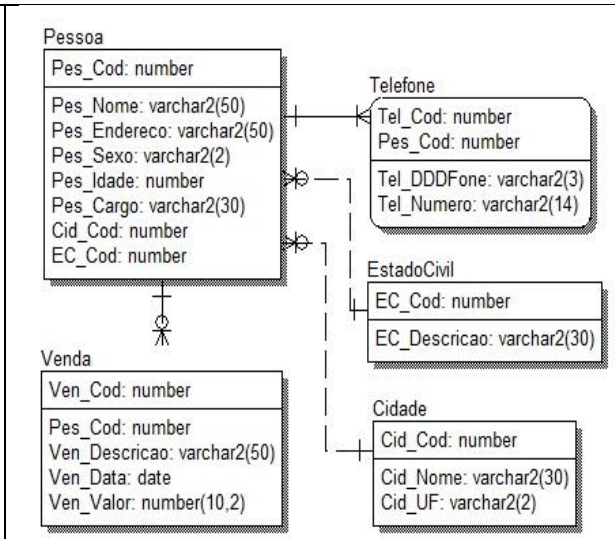


Figura 5. Base de Dados Destino - Oracle

4.3 PROCESSO DE MIGRAÇÃO

Existem alguns fatores que são levados em consideração ao realizar a migração. Entre duas bases com estruturas idênticas, é necessário verificar a base de dados do destino se está vazia ou povoada.

Com a base dados vazia, o processo de transição dos dados é trivial, pois o cuidado é durante o mapeamento, para que todos os tipos de dados da origem estejam em conformidade com os tipos compatíveis no destino e realizar a migração de forma íntegra e segura.

Com a base de dados povoada o processo de transição dos dados se torna mais complexo, além dos tipos dados da origem estarem em conformidade com os tipos do destino, podem ocorrer casos em que, na base de destino já existem dados com identificadores iguais aos da origem, assim gerando conflito de chave primária, que se não for tratado, a migração desse registro não pode ser efetivada.

Para a migração entre bases distintas, além da verificação da base de dados do destino se está vazia ou povoada, é necessário também o mapeamento de atributos diferentes, porém com equivalência, uma característica que enfatiza a ferramenta desenvolvida.

4.3.1 PLANO DE MIGRAÇÃO

Para realizar a migração é fundamental que haja o mapeamento adequado dos dados. O mapeamento relaciona conceitos similares a partir de diferentes bases entre si por uma relação de equivalência.

Para a implementação da ferramenta, foi criado um Plano de Migração, onde são definidas todas as etapas necessárias para concluir a migração. Nesse sentido o plano está organizado da seguinte forma:

- Mapeamento dos tipos de dados:
  - Verificação de Compatibilidade de:
    - Tipos de Dados
    - Limite de Tamanho dos Dados
    - Atributos obrigatórios e opcionais
  - Atributos diferentes, porém com equivalência:
    - Atributo simples na origem → Atributo chave estrangeira (FK) no destino
    - Atributo chave estrangeira (FK) na origem → Atributo simples no destino
    - Registro de entidade fraca na origem → Atributo simples no destino

- Atributo simples na origem → Gera registro de entidade fraca no destino

- Gerenciamento de Conflitos:
  - Conflito de chaves primárias:
    - Ignorar Conflitos
    - Sobrepor Valores
    - Inserir Valores
  - Atualização de chave estrangeira (FK) de acordo com suas respectivas referências no destino

4.3.2 MAPEAMENTO DOS TIPOS DE DADOS COM VERIFICAÇÃO DE COMPATIBILIDADE DOS ATRIBUTOS

Para realizar o mapeamento dos dados é preciso listar as tabelas das bases de dados origem e destino. A partir da escolha de uma tabela, seus atributos são exibidos, conforme ilustra a figura 6 em que a tabela Cliente foi escolhida e seus atributos aparecem ao lado. Para listar todas as tabelas e os atributos do SGBD origem foi realizada uma consulta nos metadados do SQL SERVER, especificamente na tabela Information_Schema.Tables para listagem das tabelas, e na tabela Information_Schema.Columns para listagem dos atributos. Para listar atributos chaves primárias (PK) são utilizadas as tabelas Information_Schema.Key_Columns e Sys.Objects. No caso dos atributos chaves estrangeiras (FK) são utilizadas as tabelas Sys.Foreing_Keys, Sys.Foreing_Key_Columns, Sys.Tables e Sys.Columns.

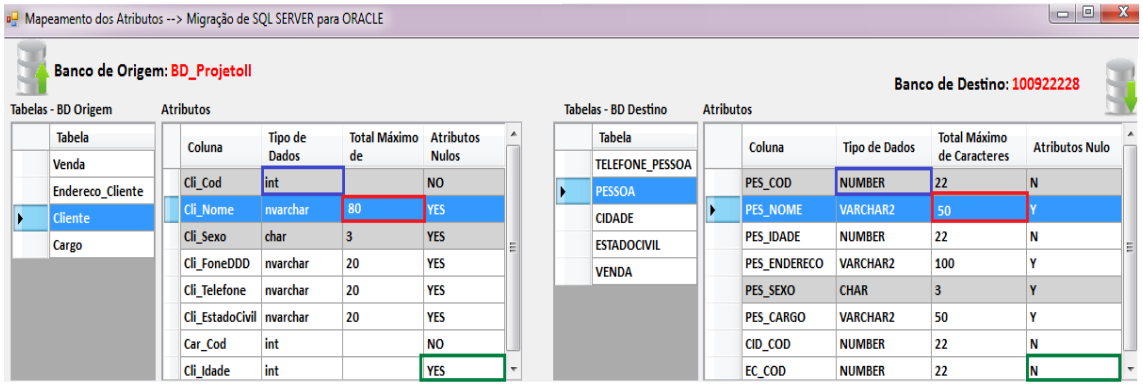
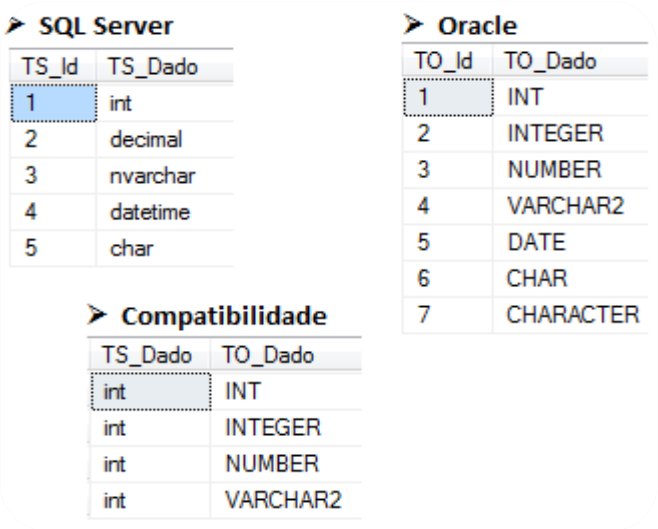


Figura 6. Listagem dos elementos - Origem e Destino

O mesmo processo é realizado para listar as informações do SGBD destino, os metadados do Oracle, onde especificamente a tabela User_Tables permite a visualização de todas as tabelas, e na tabela User_Tab_Columns permite a listagem de todos os atributos de tal tabela.Para listar atributos chaves primárias (PK) do destino são necessárias as tabelas All_Cons_Columns e All_Constraints, para listar os atributos de

chaves estrangeiras (FK) também do destino, são necessárias as estruturas All_Cons_Columns, All_Constraints e User_Constraints.

Para fazer a equivalência dos tipos de dados dos bancos SQL Server e Oracle, durante o mapeamento, foi definida uma estrutura auxiliar, contendo os tipos de dados de cada SGBD definido como origem e destino, bem como a relação de equivalência entre os tipos. A Figura 7 representa parte dessa estrutura auxiliar, que garante a equivalência dos tipos de dados, onde é possível identificar, na tabela Compatibilidade, os tipos no Oracle que são equivalentes ao tipo intno SQL Server. Com isso, a compatibilidade de tipos é garantida durante um processo de migração.



SQL Server

TS_Id	TS_Dado
1	int
2	decimal
3	nvarchar
4	datetime
5	char

Oracle

TO_Id	TO_Dado
1	INT
2	INTEGER
3	NUMBER
4	VARCHAR2
5	DATE
6	CHAR
7	CHARACTER

Compatibilidade

TS_Dado	TO_Dado
int	INT
int	INTEGER
int	NUMBER
int	VARCHAR2

Figura 7. Tabela de Compatibilidade de Tipos de Dados

Para casos de dados do tipo string além da verificação da compatibilidade dos tipos de dados, também é obrigatório que o limite de tamanho do dado do destino seja maior ou igual ao da origem, para que o mapeamento possa ser concluído. Caso contrário a ferramenta permite ao usuário atualizar o tamanho do atributo do destino de acordo com o tamanho do atributo da origem.

Por último, há o mapeamento dos dados verificando atributos opcionais na origem e obrigatórios no destino. Nesse caso, obrigatoriamente um valor padrão deve ser inserido no destino, caso contrário, a migração não pode ser realizada, pois essa restrição é garantida automaticamente pelo SGBD.

Todas as vezes que ocorrerem situações como essas, o usuário será notificado sobre o tipo de mapeamento que

está sendo gerado, cabe a ele prosseguir com o mapeamento ou não, pois é uma ferramenta de funcionamento semiautomático.

4.3.2.1 MAPEAMENTO DE ATRIBUTO SIMPLES NA ORIGEM COM ATRIBUTO CHAVE ESTRANGEIRA NO DESTINO

Considerando a tabela Cliente com o atributo Cli_EstadoCivilna base de dados origem, conforme apresentado na Figura 4, sendo mapeado para o atributo EC_Cod da tabela Pessoa da base de dados destino, onde o atributo do destino é uma chave estrangeira, como pode-se observar na Figura 5.Nesse tipo de mapeamento, a regra é selecionar na tabela de referência do EC_Cod(FK)que éa tabela EstadoCivil, qual o atributo dessa tabela tem valor correspondente ao valor que está no atributo da origem Cli_EstadoCivil, conforme apresentado na Figura 8.

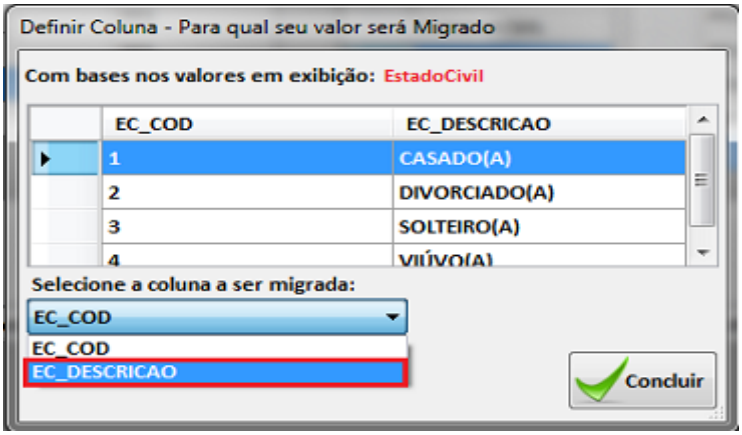


Figura 8. Processo de seleção para correspondência de valores

Logo, no processo de migração, agora sob funcionamento da ferramenta, é retornado para inserção no atributo EC_Cod(FK)da tabela Pessoa, o valor do atributo EC_Cod(PK)que referencia o registro na tabela EstadoCivil, em que o atributo EC_Descricao tem como valor o mesmo resultado do atributo Cli_EstadoCivil da tabela Cliente da origem, conforme mostra a figura 9.

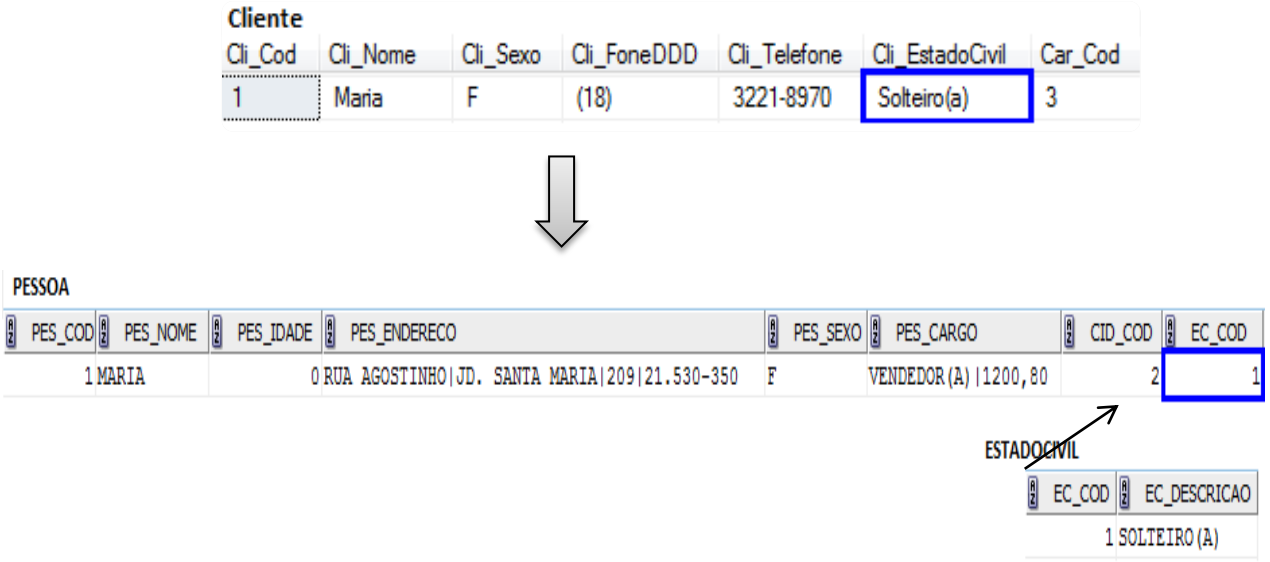


Figura 9. Resultado da migração - atributo simples na origem → atributo FK no destino

Esse tipo de mapeamento garante a integridade da informação migrada, mesmo com estruturas distintas as informações são equivalentes.

4.3.2.2 MAPEAMENTO DE ATRIBUTO CHAVE ESTRANGEIRA NA ORIGEM COM ATRIBUTO SIMPLES NO DESTINO

Considerando a tabela Cliente com o atributo Car_Cod(FK)uma chave estrangeira na base de dados origem, conforme apresentado na Figura 4, sendo mapeado para o atributo Pes_Cargo da tabela Pessoa da base de dados destino, como pode-se observar na Figura 5. Nesse tipo de mapeamento, a regra é selecionar quais os atributos da tabela Cargo que é referenciada pelo atributo Car_Cod(FK) da tabela Cliente, serão transportados para o destino, conforme apresentado na Figura 10.

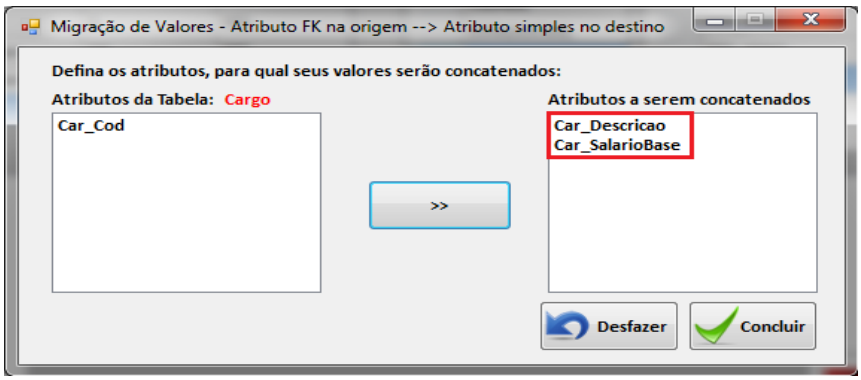


Figura 10. Seleção dos atributos, para concatenação de valores.

Nessa etapa a ferramenta permite que todos os dados do registro referenciado pelo atributo Car_Cod(FK) da tabela Cliente, que são os registros da tabela Cargo que estão na origem, sejam transportados para o destino e inseridos no atributo Pes_Cargo da tabela Pessoa. Os valores dos atributos selecionados são transportados de forma concatenada, sem que haja a perda de informações, conforme mostra a figura 11. Lembrando que nesse caso, a decisão de quais atributos serão transportados para o destino, fica sob responsabilidade do usuário.

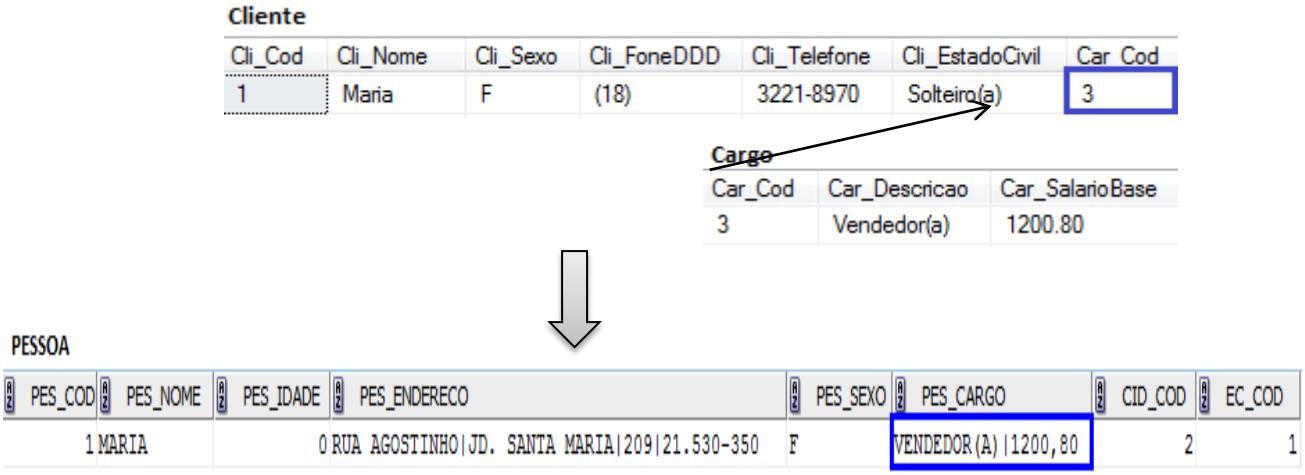


Figura 11. Resultado da migração – atributo FK na origem → atributo simples no destino

Essa regra permite a migração dos dados de modo que não haja disseminação de informações, migrando estruturas diferentes, porém com informações totalmente equivalentes.

4.3.2.3 MAPEAMENTO DE ATRIBUTOS DE ENTIDADE FRACA NA ORIGEM COM ATRIBUTO SIMPLES NO DESTINO

Considerando a tabela Endereco com o atributo End_Cod uma chave primária na base de dados origem, conforme apresentado na Figura 4, sendo mapeado para o atributo Pes_Endereco da tabela Pessoa da base de dados destino, como pode-se observar na Figura 5.Para esse tipo de mapeamento a regra definida é, onde o Cli_Cod da tabela Cliente for igual ao Cli_Cod da tabela Endereco na base de origem, migra-se o 1º endereço encontrado. Essa regra de migração foi definida em decorrência do número de endereços que um único cliente pode ter, torna-se praticamente impossível migrar mais de um endereço por cliente, pois o atributo receptor no destino é único.

O processo de migração utilizado nessa etapa é semelhante ao processo definido no tópico anterior, onde a ferramenta permite que o usuário selecione

os atributos da tabela Endereco que deseja transportar para o destino, após a seleção, de forma concatenada os valores dos atributos são inseridos no atributo Pes_Endereco da tabela Pessoa do banco de dados de destino, conforme mostra a figura 12 o resultado dessa operação.

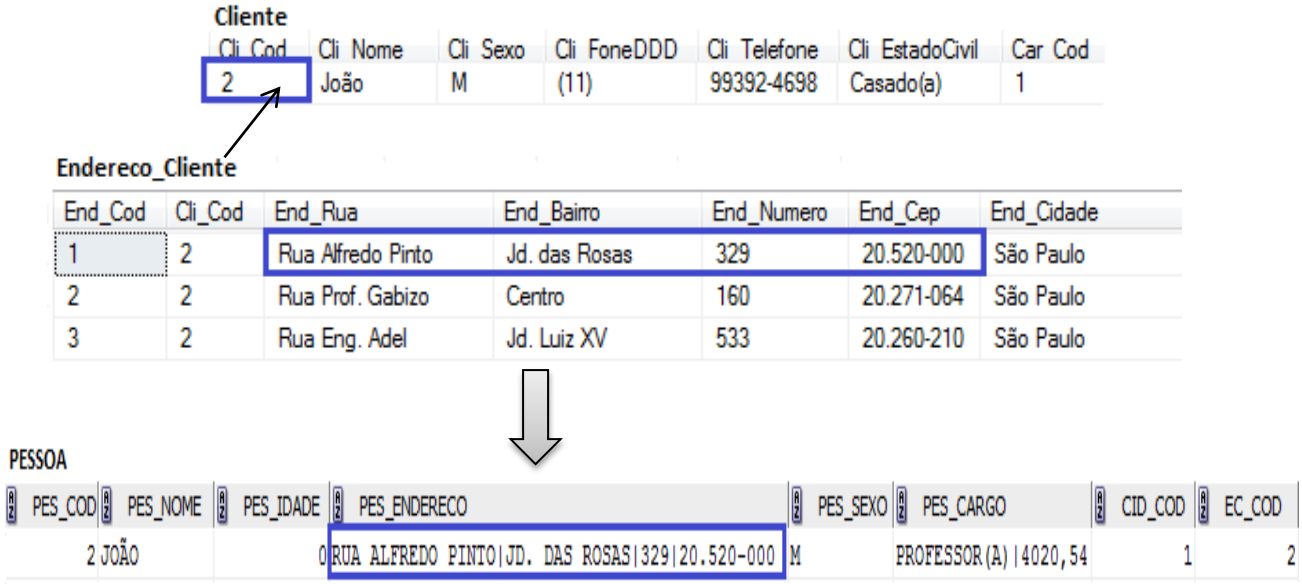


Figura 12. Resultado da migração – atributos de entidade fraca na origem → atributo simples no destino

4.3.2.4 MAPEAMENTO DE ATRIBUTO SIMPLES NA ORIGEM QUE GERA UM REGISTRO DE ENTIDADE FRACA DO DESTINO

Considerando a tabela Cliente com o atributo Cli_FoneDDD na base de dados origem, conforme apresentado na Figura 4, sendo mapeado para o atributo Tel_DDDFone da tabela Telefone da base de dados destino, como pode-se observar na Figura 5. Para esse tipo de mapeamento a regra definida é, para todo registro na origem da tabela Cliente que o Cli_FoneDDD for não nulo, insere-se um novo registro na tabela Telefone do destino, onde atributo Pes_Cod(PK) da tabela Pessoa, seja igual ao atributo Pes_Cod(FK) da tabela Telefone e o atributo Tel_DDDFone da tabela Telefone do destino, recebe o valor do atributo Cli_FoneDDD da tabela Cliente presente na origem. Esse processo se estende para todos os atributos da origem que forem mapeados com atributos da entidade fraca Telefone do destino. Gerando um novo registro na tabela Telefone do destino, conforme mostra a figura 13.

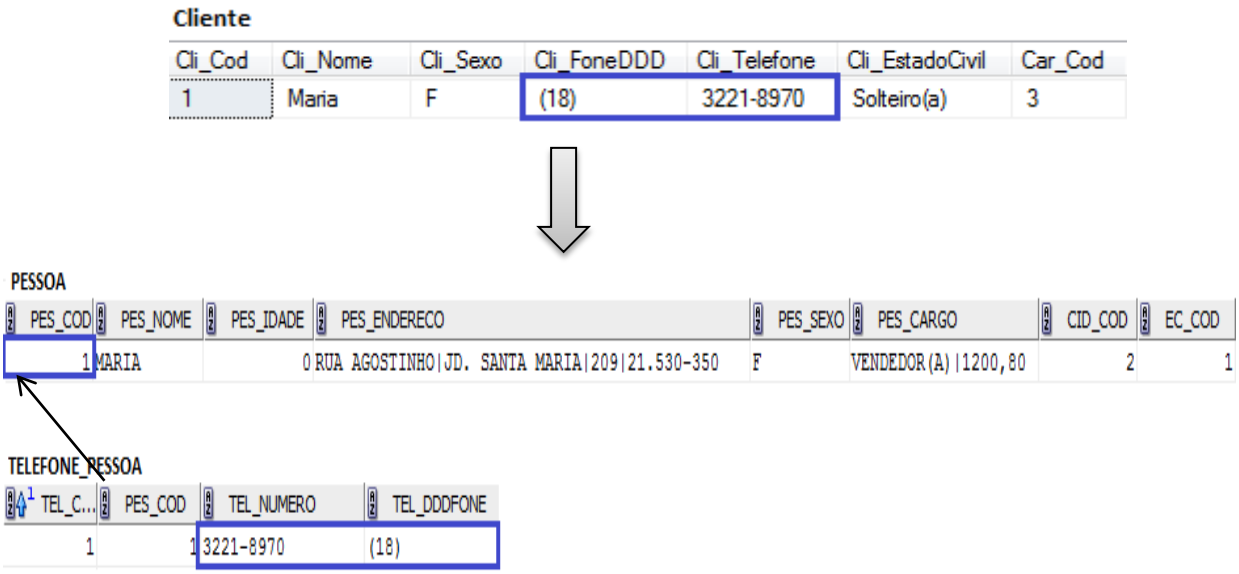


Figura 13. Resultado da migração - Atributo simples na origem → gera registro de entidade fraca no destino]

4.3.2.5 GERENCIAMENTO DE CONFLITOS

Para realizar a migração entre bases de dados independente de suas estruturas, é essencial verificar se a base de destino está vazia ou povoada.

No caso da base de dados vazia, o processo de migração é realizado respeitando apenas as regras de mapeamento.

Com relação à base de dados povoada, adicionalmente é obrigatório o cumprimento da etapa de gerenciamento de conflitos, onde podem ocorrer casos em que, na base de destino já existam dados,assim gerando conflito de chave primária, que se não for tratado, a migração desse registro não pode ser efetivada.

Para ocorrências desse tipo existem três opções de solucionar o problema, são elas:

- **Ignorar Conflito:** essa opção é direcionada a registros iguais, onde o registro da origem não é transportado para o destino. Mantém como informação o que já existe no destino.
- **Sobrepor Valores:** essa opção é direcionada a registros com identificadores iguais, mas que os outros atributos que compõem o registro tenham valores similares em comparação aos dados da origem, portanto é permitido que o registro do destino seja atualizado com os valores do registro da origem, mantendo o identificador existente no destino.
- **Inserir Valores:** essa opção é direcionada a registros que são distintos

entre si, tanto na origem como no destino, mas que o atributo identificador de ambos sejam iguais, nessa situação é permitido que o registro da origem seja inserido no destino com um novo identificador, preservando a consistência dos dados, evitando a perda de informações, algo que é inadmissível em um processo de migração de dados.

Durante essas ocorrências a escolha por uma dessas opções é responsabilidade do usuário, verificando os conflitos de forma que sejam: registros iguais, registros semelhantes ou registros diferentes com identificadores iguais. Essas comparações são realizadas durante o processo de migração, entre registros da base de origem com registros da base de destino.

Além de tratar todos os conflitos de chaves primárias (PK), por consequência a ferramenta também faz a atualização automática de chave estrangeira (FK) com suas respectivas referências no destino. Essa é mais uma das características que enfatiza a ferramenta desenvolvida.

4.4 UNIFICAÇÃO DOS DADOS

A unificação é uma forma de diminuir a carga da base de dados, eliminando informações redundantes. Durante o processo de migração, na comparação de valores com equivalência, pode ser que no destino não exista tal dado, havendo a obrigação de inseri-lo e nesse processo podem ser inseridos dados repetidos, conforme o processo ilustrado pela figura 14. Nesse caso foi inserida uma única cidade com três nomes diferentes. Portanto, para a unificação das bases o usuário define um registro, como sendo o original e faz a unificação com os outros que são equivalentes. Isso implica em alterar para a cidade 2 Presidente Prudente SP,todas as referências de chaves estrangeiras que tenham o valor 3 P. Prudente SP e 4 Pres. Prudente SP. Esse processo de unificação se estende para qualquer tabela que terá a necessidade de ser unificada.

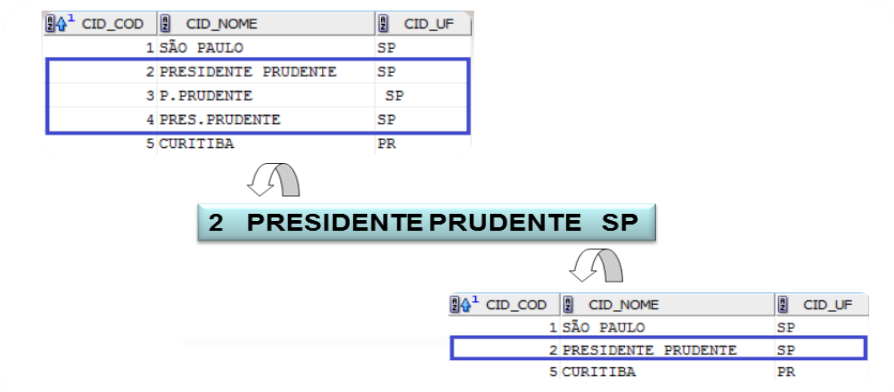


Figura 14. Unificação da base de dados

5 CONCLUSÕES

Esse trabalho apresenta uma alternativa para a migração de dados entre bancos de dados relacionais, sendo eles, SQL Server para Oracle, que visa dar suporte a empresas que possuem sistemas com banco de dados legados, e que necessitam mudar de base de dados, migrando dados/informações de antigos para novos sistemas gerenciadores de banco de dados.

A ferramenta permite facilidade de uso na migração de grandes volumes de dados de um SGBD para outro de forma semiautomática.

Possui o desempenho voltado em disponibilizar as informações do antigo para o novo SGBD, muito mais rapidamente e deixando o novo banco de dados apto a prosseguir com suas atividades normais, além da diminuição do índice de erros, sendo praticamente insignificante se comparado ao processo realizado manualmente.

Para trabalhos futuros pode ser implementada a opção de migração de tabelas com relacionamento de muitos-para-muitos (N:N) e a criação de um projeto com interface mais funcional e atraente para os processos de migração.

REFERÊNCIAS

SILBERSCHATZ, A.; KORTH, H.F.; SUDARSHAN, S. **Sistemas de banco de dados**. 3. ed. São Paulo: Makron Books, 1999.

HEUSER, C.A. **Projeto de banco de dados.** 4. ed. Porto Alegre: Bookman, 2001.

PRIMACK, F.V. **Decisões com BI (Business Intelligence).** 2. ed. 2008.

MONTEIRO, J.M.; LIFSCHITZ, S.; BRAYNER, A. **Extraindo metadados de SGBDs.** 2007. Monografia (Ciência da Computação) - Pontifícia Católica do Rio de Janeiro.

NETO, J.R.; PASSOS E.B. **JExodus:** uma ferramenta para migração de dados independente de SGBD. Teresina: Instituto Federal de Educação e Ciência e Tecnologia do Piauí, 2009.

BARRETTO, R.L.M. **Data migration wizard: um assistente de migração de dados para banco de dados no SQL Server.** Recife: Centro de Informática, Universidade Federal de Pernambuco, 2008.