

Jackson Júnior

Leonardo Lopez

**Projeto: integração de fluxos de informação (Sanitop 360)**

Guia de implementação

Viana do Castelo

2022

Jackson Barreto Costa Júnior

Leonardo MAriz Martinez Lopez

**Projeto: integração de fluxos de informação (Sanitop 360)**

Guia de implementação

.

Guia de implementação sobre o trabalho realizado através do estágio extracurricular na empresa Sanitop no curso de Licenciatura em Engenharia Informática do Instituto Politécnico de Viana do Castelo – IPVC.

Orientação: Professor doutor Pedro Coutinho, Professora doutora Sara Paiva, e Professor doutor Ricardo Freitas.

Viana do Castelo

2022

**SUMÁRIO**

[1. Introdução 5](#_Toc98171474)

[2. Planeamento da arquitetura 5](#_Toc98171475)

[3. Implementação 11](#_Toc98171476)

[3.1. Configuração dos nodos 12](#_Toc98171480)

[3.1.1. engine.name 12](#_Toc98171481)

[3.1.2. external.id 12](#_Toc98171482)

[3.1.3. group.id 12](#_Toc98171483)

[3.1.4. sync.url 12](#_Toc98171484)

[3.1.5. registration.url 12](#_Toc98171485)

[3.1.6. db.driver 13](#_Toc98171486)

[3.1.7. db.url 13](#_Toc98171487)

[3.1.8. db.user 13](#_Toc98171488)

[3.1.9. db.password 13](#_Toc98171489)

[3.2. Configuração da sincronização 14](#_Toc98171490)

[3.2.1. Links de comunicação 16](#_Toc98171491)

[3.2.2. Definindo Rotas 17](#_Toc98171492)

[3.2.3. Canais 18](#_Toc98171493)

[3.2.4. Definindo Triggers 19](#_Toc98171494)

[3.2.5. Roteamento dos Triggers 20](#_Toc98171495)

[3.2.6. Transformações 21](#_Toc98171496)

[4. Anexo 25](#_Toc98171497)

[4.1. Script SQL de instalação 25](#_Toc98171499)

[4.1.1. Configuração dos Nodos 25](#_Toc98171500)

[4.1.2. Configuração dos links. 26](#_Toc98171501)

[4.1.3. Configuração do roteamento 27](#_Toc98171502)

[4.1.4. Configuração dos canais 28](#_Toc98171503)

[4.1.5. Configuração dos triggers 29](#_Toc98171504)

[4.1.6. Configuração do roteamento dos triggers 30](#_Toc98171505)

[4.1.7. Transformação de tabelas 31](#_Toc98171506)

[4.1.8. Transformação de colunas 42](#_Toc98171507)

1. Introdução

O presente guia de implementação tem por objetivo servir como orientação para a implementação do sistema de sincronização das bases de dados da empresa **Sanitop**, através do software **SymmetricDS**.

O guia discorrerá sobre as etapas necessárias para se realizar a integração de fluxos de informação através do SymmetricDS, considerando o contexto do ambiente de produção da Sanitop.

É importante ressaltar que este documento não tem a pretensão de substituir a documentação oficial do SymmetricDS, consistindo apenas em um relatório sobre pesquisas empíricas sob as diretrizes da documentação oficial. Portanto, para esclarecimentos mais pontuais, recomenda-se vivamente revisar a documentação oficial.

1. Planeamento da arquitetura

Em uma arquitetura de solução de integração de fluxo de informações, é importante identificar o sentido em que os dados devem fluir; as redes e servidores nos quais as bases de dados estão provisionadas; e as versões dos SGBDs (Sistema de Gerenciamento de Base de Dados) em execução.

Diagrama

Descrição gerada automaticamente

Figura 1 - Topologia e fluxo de informações.

A Figura 1 mostra a rede em que cada base de dados está provisionada, assim como o fluxo de informações/dados, isto é, o sentido no qual a informação precisa fluir. Note que entre os servidores SRVA e SRVB há um canal de dados bidirecional, enquanto as duas instâncias de base de dados que estão executando no mesmo servidor SRVC (situação incomum) enviam dados para o SRVB por um canal unidirecional.

A identificação das redes é extremamente útil na fase de implementação para definição das políticas de segurança da rede, como quais portas devem ser abertas nos Firewalls.

Na sequência, deve ser elaborado um mapa de correlação das tabelas a serem sincronizadas, ou seja, identificar as tabelas da origem que precisam ser sincronizadas com as tabelas do destino. Isto é necessário porque uma tabela de origem pode ter um nome diferente em seu destino e conter apenas algumas das colunas da origem.

|  |  |  |
| --- | --- | --- |
| **CORRELAÇÃO DE TABELAS** | | |
| **#** | **Base de Dados 03** | **Base de Dados 02** |
| 01 | vendas | vendas\_centralizada |
| 02 | pedidos | confirmados |
| 03 | retomas | retomas |

Tabela 1 - Correlação de tabelas

A Tabela 1 exemplifica uma sincronização entre a *base de dados 03 e a 02*, onde as tabelas *vendas* e *pedidos* tem seu nome alterado, enquanto a tabela *retoma* mantém o nome.

|  |  |  |
| --- | --- | --- |
| **CORRELAÇÃO DE COLUNAS** | | |
| **#** | **Base de Dados 03 - vendas** | **Base de Dados 02 – vendas\_centralizada** |
| 01 | qty | despachos |
| 02 | date | ocorrencia |
| **#** | **Base de Dados 03 - pedidos** | **Base de Dados 02 – confirmados** |
| 01 | cliente | associado |
| 02 | data | data |

Tabela 2 - Correlação de colunas

Já a Tabela 2 mapeia as colunas que sofrem alteração de nome ou que devem ser consideradas na operação. Observe que a tabela *retomas* não foi citada, o que significa que esta tabela será sincronizada integralmente, ou seja, terá todas as suas colunas consideradas.

Outra informação relevante levantada nessa fase preliminar é a versão do SGBD em execução na base de dados. Com essa informação, pode-se definir corretamente os drives JDBC necessários para a implementação.

Cabe salientar que o SymmetricDS vem acompanhado com alguns drives JDBC, entretanto, pode ocorrer alguma incompatibilidade entre as versões em execução e as disponibilizadas pelo aplicativo, como falhas de autenticação ou reconexões excessivas. Justamente por esse possível problema é importante documentar previamente a versão do SGBD para selecionar o drive correto.

|  |  |  |
| --- | --- | --- |
| **REGISTO DE SGBD** | | |
| **Base de Dados** | **SGBD** | **Versão** |
| Base de Dados 01 | Microsoft SQL Server 2019 | 15.2.1 |
| Base de Dados 02 | Postgre SQL 9.6 | 9.6.24 |
| Base de Dados 03 | Microsoft SQL Server 2016 | 12.5.7 |

Tabela 3 - Registo de SGBD

A Tabela 3 identifica o SGBD e a versão corrente de cada uma das bases de dados que compõem o processo de integração.

Finalmente, deve-se definir um agrupamento transacional, que tem como finalidade identificar todas as tabelas que fazem parte de uma mesma transação, mesmo aquelas que não estão relacionadas por chave estrangeira, mas sim por uma regra de negócio que gera uma transação.

Essa informação é importante para agrupar essas tabelas no mesmo **canal de comunicação**, caso contrário ocorrerão erros no momento de replicar os dados, por inconsistência dos dados.

|  |  |
| --- | --- |
| **REGISTO DE TRANSAÇÕES** | |
| **Base de dados 01** | |
| **Transação 01** | |
| **#** | **Tabelas** |
| 01 | qty |
| 02 | date |

Tabela 4 - Registo de transações

A Tabela 4 demonstra que as tabelas *qty* e *date* fazem parte de uma mesma transação, e por isso devem ser sincronizadas em conjunto, de modo a manter o seu relacionamento e integridade dos dados.

Por fim deve ser identificado os agrupamentos de base de dados.

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Base de dado** | **Porta** | **Grupo** |
| 01 | Filial01 | 8443 | Filial |
| 02 | Filial02 | 8443 | Filial |
| 03 | Filial03 | 8443 | Filial |
| 04 | Filial04 | 8443 | Filial |
| 05 | Central | 8443 | Central |
| 06 | financeiro | 8443 | Financeiro |

Tabela 5 - Agrupamento de base de dado

A Tabela 5 permite-nos observar que as bases de dados *filial01*, *filial02*, *filial03* e *filial04* pertencem ao mesmo grupo, enquanto a *central* e *financeiro* estão individualizadas.

As bases de dados das filiais não necessitam de estar no mesmo servidor, para estarem no mesmo agrupamento (como se observa pelo uso da mesma porta TCP). O agrupamento se justifica pelo contexto, ou seja, todas as filiais devem receber as mesmas atualizações de dados da central. A definição, neste caso, é que a central fornece dados para todas as suas filiais.

Reunidas todas estas informações o projeto encontra-se apto a seguir para a fase de implementação.

1. Implementação
   1. Instalação

A instalação do SymmetricDS consiste em, após baixado o ficheiro ZIP no site oficial, descompactar do ficheiro na diretoria onde instalar o SymmetricDS.

Habitualmente a porta TCP padrão, utilizada pelo SymmetricDS, não é utilizada por outras aplicações, em caso de necessidade de alterar a porta TCP basta aceder ao ficheiro conf/symmetric-server.properties , a partir da diretoria de instalação.

Texto

Descrição gerada automaticamente

Figura 2 - Configuração da porta de acesso

A Figura 2 exibe o atributo para configurar a porta TCP na qual o SymmetricDS irá operar. Destaca-se que a porta TCP definida deve ser informada nos ficheiros de configuração, como parte da URL de registro e sincronização, conforme demonstrado no tópico seguinte (configuração dos nodos).

Por fim, deve-se configurar o SymmetricDS para ser executado como um serviço do Windows. Para tal, deve-se executar, **com permissões de administrador**, o ficheiro bin/sym\_service.bat install , a partir da diretoria de instalação.

Desta forma, o serviço de sincronização iniciará automaticamente quando o sistema inicializar, sendo executado como um serviço do Windows. Um processo de *wrapper* inicia o SymmetricDS e o monitora, para que possa ser reiniciado se ficar sem memória ou sair inesperadamente.

* 1. Configuração dos nodos

De modo a obter a melhor performance o SymmetricDS deve ser instalado no mesmo servidor onde está a base de dado a ser monitorada.

Dentro do contexto do SymmetricDS, cada base de dado é representada como um nodo. O nodo é definido por um ficheiro de extensão “*properties*” que deve ser armazenado na diretoria “*engine/*”, e possui as seguintes configurações obrigatórias:

* + 1. engine.name

É o nome utilizado para acessar o servidor SymmetricDS deste nodo, através da URL de comunicação.

No caso do engine.name ser *DynamanWMS* a URL de acesso seria: http://srvsql:8443/sync/DynamanWMS

* + 1. external.id

É um nome lógico, que possui significado no contexto de negócio onde está sendo implementado, como *filial01* ou *financeiro*.

* + 1. group.id

O grupo de nós do qual esse nó é membro, para esta configuração basta recorrer a sua tabela de agrupamento de base de dados (Tabela 5).

* + 1. sync.url

É a URL utilizada pelos servidores SymmetricDS para comunicação. A URL de sincronização tem o formato: http://{hostname}:{port}/sync/{engine.name}

* + 1. registration.url

Esta é a URL do servidor de configuração SymmetricDS, que é o servidor de registro onde se encontram as configurações de sincronização de todo o projeto.

Cada nodo se registra neste servidor e passa a receber as configurações e atualizações para o funcionamento do sistema de sincronização.

* + 1. db.driver

O nome da classe Java do driver JDBC.

Exemplo: com.microsoft.sqlserver.jdbc.SQLServerDriver

* + 1. db.url

A URL JDBC usada para se conectar ao banco de dados.

* + 1. db.user

O nome de usuário do banco de dados, que é usado para efetuar login, criar e atualizar tabelas. Recomenda-se que seja criado um utilizador exclusivamente para o SymmetricDS.

* + 1. db.password

A senha para o usuário do banco de dados.

Tela preta com letras brancas

Descrição gerada automaticamente

Figura 3 - Exemplo de configuração de nodo

A Figura 3 apresenta um ficheiro de configuração de um nodo SymmetricDS, com a configuração mínima necessária para sua execução.

* 1. Configuração da sincronização

O primeiro passo para configurar a sincronização é criar os nodos propriamente ditos na base de dados do servidor de registro. Isso irá permitir que todos os nodos do sistema recebam as configurações que lhes dizem respeito.

Por isso, os scripts SQL que configuram o SymmetricDS devem ser executados na base de dados do nodo mestre (servidor de registro).

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

Figura 4 - Configuração dos nodos

A Figura 4 exibe um código SQL para configuração dos nodos.

Observe que o os campos *create\_time*, *last\_update\_time* e *last\_update\_by*, apesar de serem opcionais, são de extrema importância para manter o sistema auditável. Em nosso exemplo utilizamos o valor “*installation*” para representar que estas definições foram realizadas na instalação.

* + 1. Links de comunicação

Realizada a definição dos nodos é o momento de configurar os links de comunicação entre eles. Nesta etapa se está a implementar as definições estabelecidas na Figura 1, ou seja, os fluxos de comunicação.

Texto

Descrição gerada automaticamente

Figura 5 - Configuração dos Links

Os links definem em alto nível como os dados se movem em todo o cenário de sincronização. O link define quais **grupos de nós** sincronizarão dados com outros grupos de nós e, dentro dessa troca, qual grupo de nós iniciará a conversa para essa troca.

Um canal pode estar configurado, em “*data\_event\_action*”, como “P” para “*push*” e “W” para “*pull*”.

O “*push*” indica que os nodos do grupo de nodos de origem irão iniciar a comunicação, ou seja, enviarão dados para os nodos de destino.

Enquanto que “*pull*” sinaliza que os nodos de origem irão aguardar um nodo de destino iniciar a comunicação, isto também permite que os nodos de destino extraiam dados dos nodos de origem.

* + 1. Definindo Rotas

Texto

Descrição gerada automaticamente

Figura 6 - Configuração de roteamento

* + 1. Canais

Depois que os links de grupo e os roteadores são definidos, é o momento de especificar quais dados devem ser sincronizados nesses links e roteadores. Os canais definem os agrupamentos lógicos realizados na Tabela 4, ou seja, os registros de transações.

Texto

Descrição gerada automaticamente

Figura 7 - Configuração dos Canais

Na Figura 7 todos os dados serão transmitidos no mesmo canal. No entanto, quando se possui operações que demandam uma alta carga de dados, como blobs de grandes dimensões, recomenda-se que estes (e os associados à mesma transação) sejam transmitidos em um canal separado, para otimizar a performance de transferência de dados.

* + 1. Definindo Triggers

Quando os triggers são definidos no SymmetricDS, eles são automaticamente gerados na base de dados correspondente.

Texto

Descrição gerada automaticamente

Figura 8 - Configuração dos Triggers

A Figura 8 demonstra uma configuração de *triggers* para várias tabelas de uma só vez. Todas as alterações nestas tabelas serão capturadas, ainda que uma coluna tenha seu valor atualizado para o mesmo valor, basta que ocorra uma operação de “*update*” para que o *trigger* seja disparado, embora esse comportamento padrão possa ser alterado.

* + 1. Roteamento dos Triggers

A configuração de gatilhos e roteadores é uma relação muitos-para-muitos que define quais combinações específicas de gatilhos e roteadores são necessárias para a configuração. Essa tabela serve como uma tabela de junção para definir quais combinações são válidas, bem como para definir as configurações disponíveis no nível de granularidade do roteador do gatilho.

Texto

Descrição gerada automaticamente

Figura 9 – Roteamento dos triggers

* + 1. Transformações

Em alguns casos, pode ser necessário realizar algumas transformações, como alterar o nome da tabela de origem no destino ou montar a tabela de destino baseada em mais de uma tabela de origem.

Texto

Descrição gerada automaticamente

Figura 10 - Alterando o nome da tabela

A Figura 10 demonstra a configuração para alterar o nome de uma tabela em seu destino. A coluna “*transform\_point*” define se a transformação proposta acontece na origem ou no destino, a princípio esta transformação ocorrerá na origem, segundo a metodologia clássica ETL (Extract Transform Load).

Texto

Descrição gerada automaticamente

Figura 11 - transformação de coluna

A Figura 11 demonstra a transformação de uma coluna onde simplesmente se copiam os dados de uma coluna de origem para a coluna de destino, sem nenhuma alteração. Este procedimento é necessário quando se copia apenas alguma das colunas de uma tabela e não todas elas implicitamente.

Texto

Descrição gerada automaticamente

Figura 12 - Transformação de coluna de tabela terceira

Já a Figura 12 demonstra que a coluna de destino não será copiada de nenhuma coluna na tabela atual, mas sim de uma tabela “*lookup*” e o dado a ser selecionado nesta tabela é obtido através da instrução SQL definida no campo “*transform\_expression*”.

Texto

Descrição gerada automaticamente

Figura 13 - Expressões complexas em transformação

A Figura 13 demonstra que expressões SQL diversas podem ser utilizadas, de modo a decompor um script em várias instruções menores.

1. Anexo
2. 1. Script SQL de instalação
      1. Configuração dos Nodos

-- CONFIGURAÇÃO DOS NODOS

INSERT INTO SYM\_NODE\_GROUP

(

    node\_group\_id,

    description,

    create\_time,

    last\_update\_time,

    last\_update\_by

)

VALUES

(

    'DynamanWMS',

    'IBS Dynaman WMS - SQL Server',

    current\_timestamp,

    current\_timestamp,

    'Installation'

);

INSERT INTO SYM\_NODE\_GROUP

(

    node\_group\_id,

    description,

    create\_time,

    last\_update\_time,

    last\_update\_by

)

VALUES

(

    'Sanitop360',

    'Sanitop 360 - SQL Server',

    current\_timestamp,

  current\_timestamp,

    'Installation'

);

* + 1. Configuração dos links

INSERT INTO SYM\_NODE\_GROUP\_LINK

(

    source\_node\_group\_id,

    target\_node\_group\_id,

    data\_event\_action,

    create\_time,

    last\_update\_time,

    last\_update\_by

)

VALUES

(

    'DynamanWMS',

    'Sanitop360',

    'P',

    current\_timestamp,

    current\_timestamp,

    'Installation'

);

INSERT INTO SYM\_NODE\_GROUP\_LINK

(

    source\_node\_group\_id,

    target\_node\_group\_id,

    data\_event\_action,

    create\_time,

    last\_update\_time,

    last\_update\_by

)

VALUES

(

    'Sanitop360',

    'DynamanWMS',

    'W',

    current\_timestamp,

    current\_timestamp,

    'Installation'

);

* + 1. Configuração do roteamento

INSERT INTO SYM\_ROUTER

(

    router\_id,

    source\_node\_group\_id,

    target\_node\_group\_id,

    create\_time,

    last\_update\_time,

    last\_update\_by,

    description

)

VALUES

(

    'DynamanWMS-2-Sanitop360',

    'DynamanWMS',

    'Sanitop360',

    current\_timestamp,

    current\_timestamp,

    'Installation',

    'Router DynamanWMS to Sanitop360'

);

* + 1. Configuração dos canais

INSERT INTO SYM\_CHANNEL

(

    channel\_id,

    processing\_order,

    max\_batch\_size,

    max\_batch\_to\_send,

    extract\_period\_millis,

    batch\_algorithm,

    enabled,

    reload\_flag,

    last\_update\_time,

    create\_time,

    last\_update\_by,

    description

)

VALUES

(

    'chanelDynaman',

    2,

    1000,

    10,

    0,

    'default',

    1,

    1,

    current\_timestamp,

    current\_timestamp,

    'Installation',

    'Dynaman WMS data chanel'

);

* + 1. Configuração dos triggers

INSERT INTO SYM\_TRIGGER

(

    trigger\_id,

    source\_table\_name,

    channel\_id,

    last\_update\_time,

    create\_time,

    last\_update\_by,

    description

)

VALUES

(

    'tablesDynaman',

    'task\_detail,stock\_product,picklist\_detail\_order,Container\_Inventory,outbound\_order\_header,outbound\_Order\_Detail,inbound\_Order\_Header,arrival\_detail,arrival\_header,inbound\_Order\_detail,outbound\_Order\_text',

    'chanelDynaman',

    current\_timestamp,

    current\_timestamp,

    'Installation',

    'Dynaman WMS table mapping.'

);

* + 1. Configuração do roteamento dos triggers

INSERT INTO SYM\_TRIGGER\_ROUTER

(

    trigger\_id,

    router\_id,

    initial\_load\_order,

    create\_time,

    last\_update\_time,

    last\_update\_by,

    description

)

VALUES

(

    'tablesDynaman',

    'DynamanWMS-2-Sanitop360',

    1,

    current\_timestamp,

    current\_timestamp,

    'Installation',

    'router for Dynaman WMS tables.'

);

* + 1. Transformação de tabelas

INSERT INTO sym\_transform\_table

(

    transform\_id,

    source\_node\_group\_id,

    target\_node\_group\_id,

    transform\_point,

    source\_table\_name,

    target\_table\_name,

    delete\_action,

    update\_action,

    column\_policy,

    create\_time,

    last\_update\_time,

    last\_update\_by,

    description

)

VALUES

(

    'taskDetailChangeName',

    'DynamanWMS',

    'Sanitop360',

    'EXTRACT',

    'task\_detail',

    'Dyn56\_task\_detail',

    'DEL\_ROW',

    'UPD\_ROW',

    'IMPLIED',

    current\_timestamp,

    current\_timestamp,

    'Installation',

    'Changes the name of the source table on the target.'

);

INSERT INTO sym\_transform\_table

(

    transform\_id,

    source\_node\_group\_id,

    target\_node\_group\_id,

    transform\_point,

    source\_table\_name,

    target\_table\_name,

    delete\_action,

    update\_action,

    column\_policy,

    create\_time,

    last\_update\_time,

    last\_update\_by,

    description

)

VALUES

(

    'stockProductChangeName',

    'DynamanWMS',

    'Sanitop360',

    'EXTRACT',

    'stock\_product',

    'Dyn56\_stock\_product',

    'DEL\_ROW',

    'UPD\_ROW',

    'IMPLIED',

    current\_timestamp,

    current\_timestamp,

    'Installation',

    'Changes the name of the source table on the target.'

);

INSERT INTO sym\_transform\_table

(

    transform\_id,

    source\_node\_group\_id,

    target\_node\_group\_id,

    transform\_point,

    source\_table\_name,

    target\_table\_name,

    delete\_action,

    update\_action,

    column\_policy,

    create\_time,

    last\_update\_time,

    last\_update\_by,

    description

)

VALUES

(

    'picklistDetailOrderChangeName',

    'DynamanWMS',

    'Sanitop360',

    'EXTRACT',

    'picklist\_detail\_order',

    'Dyn56\_picklist\_detail\_order',

    'DEL\_ROW',

    'UPD\_ROW',

    'IMPLIED',

    current\_timestamp,

    current\_timestamp,

    'Installation',

    'Changes the name of the source table on the target.'

);

INSERT INTO sym\_transform\_table

(

    transform\_id,

    source\_node\_group\_id,

    target\_node\_group\_id,

    transform\_point,

    source\_table\_name,

    target\_table\_name,

    delete\_action,

    update\_action,

    column\_policy,

    create\_time,

    last\_update\_time,

    last\_update\_by,

    description

)

VALUES

(

    'outboundOrderDetailChangeName',

    'DynamanWMS',

    'Sanitop360',

    'EXTRACT',

    'outbound\_Order\_detail',

    'Dyn56\_Outbound\_order\_Detail',

    'DEL\_ROW',

    'UPD\_ROW',

    'IMPLIED',

    current\_timestamp,

    current\_timestamp,

    'Installation',

    'Changes the name of the source table on the target.'

);

INSERT INTO sym\_transform\_table

(

    transform\_id,

    source\_node\_group\_id,

    target\_node\_group\_id,

    transform\_point,

    source\_table\_name,

    target\_table\_name,

    delete\_action,

    update\_action,

    column\_policy,

    create\_time,

    last\_update\_time,

    last\_update\_by,

    description

)

VALUES

(

    'outboundOrderHeaderChangeName',

    'DynamanWMS',

    'Sanitop360',

    'EXTRACT',

    'outbound\_order\_header',

    'Dyn56\_Outbound\_order\_header',

    'DEL\_ROW',

    'UPD\_ROW',

    'IMPLIED',

    current\_timestamp,

    current\_timestamp,

    'Installation',

    'Changes the name of the source table on the target.'

);

INSERT INTO sym\_transform\_table

(

    transform\_id,

    source\_node\_group\_id,

    target\_node\_group\_id,

    transform\_point,

    source\_table\_name,

    target\_table\_name,

    delete\_action,

    update\_action,

    column\_policy,

    create\_time,

    last\_update\_time,

    last\_update\_by,

    description

)

VALUES

(

    'inboundOrderHeaderChangeName',

    'DynamanWMS',

    'Sanitop360',

    'EXTRACT',

    'inbound\_Order\_Header',

    'Dyn56\_Inbound\_order\_Header',

    'DEL\_ROW',

    'UPD\_ROW',

    'IMPLIED',

    current\_timestamp,

    current\_timestamp,

    'Installation',

    'Changes the name of the source table on the target.'

);

INSERT INTO sym\_transform\_table

(

    transform\_id,

    source\_node\_group\_id,

    target\_node\_group\_id,

    transform\_point,

    source\_table\_name,

    target\_table\_name,

    delete\_action,

    update\_action,

    column\_policy,

    create\_time,

    last\_update\_time,

    last\_update\_by,

    description

)

VALUES

(

    'arrivalDetailChangeName',

    'DynamanWMS',

    'Sanitop360',

    'EXTRACT',

    'arrival\_detail',

    'Dyn56\_Arrival\_Detail',

    'DEL\_ROW',

    'UPD\_ROW',

    'IMPLIED',

    current\_timestamp,

    current\_timestamp,

    'Installation',

    'Changes the name of the source table on the target.'

);

INSERT INTO sym\_transform\_table

(

    transform\_id,

    source\_node\_group\_id,

    target\_node\_group\_id,

    transform\_point,

    source\_table\_name,

    target\_table\_name,

    delete\_action,

    update\_action,

    column\_policy,

    create\_time,

    last\_update\_time,

    last\_update\_by,

    description

)

VALUES

(

    'arrivalHeaderChangeName',

    'DynamanWMS',

    'Sanitop360',

    'EXTRACT',

    'arrival\_header',

    'Dyn56\_Arrival\_Header',

    'DEL\_ROW',

    'UPD\_ROW',

    'IMPLIED',

    current\_timestamp,

    current\_timestamp,

    'Installation',

    'Changes the name of the source table on the target.'

);

INSERT INTO sym\_transform\_table

(

    transform\_id,

    source\_node\_group\_id,

    target\_node\_group\_id,

    transform\_point,

    source\_table\_name,

    target\_table\_name,

    delete\_action,

    update\_action,

    column\_policy,

    create\_time,

    last\_update\_time,

    last\_update\_by,

    description

)

VALUES

(

    'inboundOrderDetailChangeName',

    'DynamanWMS',

    'Sanitop360',

    'EXTRACT',

    'inbound\_Order\_detail',

    'Dyn56\_Inbound\_Order\_Detail',

    'DEL\_ROW',

    'UPD\_ROW',

    'IMPLIED',

    current\_timestamp,

    current\_timestamp,

    'Installation',

    'Changes the name of the source table on the target.'

);

INSERT INTO sym\_transform\_table

(

    transform\_id,

    source\_node\_group\_id,

    target\_node\_group\_id,

    transform\_point,

    source\_table\_name,

    target\_table\_name,

    delete\_action,

    update\_action,

    column\_policy,

    create\_time,

    last\_update\_time,

    last\_update\_by,

    description

)

VALUES

(

    'outboundOrderTextChangeName',

    'DynamanWMS',

    'Sanitop360',

    'EXTRACT',

    'outbound\_Order\_text',

    'Dyn56\_Outbound\_Order\_Text',

    'DEL\_ROW',

    'UPD\_ROW',

    'IMPLIED',

    current\_timestamp,

    current\_timestamp,

    'Installation',

    'Changes the name of the source table on the target.'

 );

INSERT INTO sym\_transform\_table

(

    transform\_id,

    source\_node\_group\_id,

    target\_node\_group\_id,

    transform\_point,

    source\_table\_name,

    target\_table\_name,

    delete\_action,

    update\_action,

    column\_policy,

    create\_time,

    last\_update\_time,

    last\_update\_by,

    description

)

VALUES

(

    'Container\_InventoryChangeName',

    'DynamanWMS',

    'Sanitop360',

    'EXTRACT',

    'Container\_Inventory',

    'Dyn56\_Container\_Inventory',

    'DEL\_ROW',

    'UPD\_ROW',

    'SPECIFIED',

    current\_timestamp,

    current\_timestamp,

    'Installation',

    'Changes the name of the source table on the target.'

 );

* + 1. Transformação de colunas

INSERT INTO SYM\_TRANSFORM\_COLUMN

 (

        transform\_id,

        include\_on,

        target\_column\_name,

        source\_column\_name,

        pk,

        transform\_type,

        transform\_expression,

        transform\_order,

        last\_update\_time,

        last\_update\_by,

        create\_time,

        description

)

VALUES

(

        'Container\_InventoryChangeName',

        '\*',

        'location\_code',

        'location\_code',

        0,

        'copy',

        '',

        1,

        current\_timestamp,

        'Installation',

        current\_timestamp,

        'simple copy from source column to destination column.'

);

INSERT INTO SYM\_TRANSFORM\_COLUMN

 (

        transform\_id,

        include\_on,

        target\_column\_name,

        source\_column\_name,

        pk,

        transform\_type,

        transform\_expression,

        transform\_order,

        last\_update\_time,

        last\_update\_by,

        create\_time,

        description

)

VALUES

(

        'Container\_InventoryChangeName',

        '\*',

        'Container\_code',

        'container\_code',

        1,

        'copy',

        '',

        1,

        current\_timestamp,

        'Installation',

        current\_timestamp,

        'simple copy from source column to destination column.'

);

INSERT INTO SYM\_TRANSFORM\_COLUMN

 (

        transform\_id,

        include\_on,

        target\_column\_name,

        source\_column\_name,

        pk,

        transform\_type,

        transform\_expression,

        transform\_order,

        last\_update\_time,

        last\_update\_by,

        create\_time,

        description

)

VALUES

(

        'Container\_InventoryChangeName',

        '\*',

        'inventory\_qty',

        'inventory\_qty',

        0,

        'copy',

        '',

        1,

        current\_timestamp,

        'Installation',

        current\_timestamp,

        'simple copy from source column to destination column.'

);

INSERT INTO SYM\_TRANSFORM\_COLUMN

 (

        transform\_id,

        include\_on,

        target\_column\_name,

        source\_column\_name,

        pk,

        transform\_type,

        transform\_expression,

        transform\_order,

        last\_update\_time,

        last\_update\_by,

        create\_time,

        description

)

VALUES

(

        'Container\_InventoryChangeName',

        '\*',

        'qty\_on\_picklist',

        'qty\_on\_picklist',

        0,

        'copy',

        '',

        1,

        current\_timestamp,

        'Installation',

        current\_timestamp,

        'simple copy from source column to destination column.'

);

INSERT INTO SYM\_TRANSFORM\_COLUMN

 (

        transform\_id,

        include\_on,

        target\_column\_name,

        source\_column\_name,

        pk,

        transform\_type,

        transform\_expression,

        transform\_order,

        last\_update\_time,

        last\_update\_by,

        create\_time,

        description

)

VALUES

(

        'Container\_InventoryChangeName',

        '\*',

        'qty\_not\_on\_picklist',

        'qty\_not\_on\_picklist',

        0,

        'copy',

        '',

        1,

        current\_timestamp,

        'Installation',

        current\_timestamp,

        'simple copy from source column to destination column.'

);

INSERT INTO SYM\_TRANSFORM\_COLUMN

 (

        transform\_id,

        include\_on,

        target\_column\_name,

        source\_column\_name,

        pk,

        transform\_type,

        transform\_expression,

        transform\_order,

        last\_update\_time,

        last\_update\_by,

        create\_time,

        description

)

VALUES

(

        'Container\_InventoryChangeName',

        '\*',

        'picked\_qty',

        'picked\_qty',

        0,

        'copy',

        '',

        1,

        current\_timestamp,

        'Installation',

        current\_timestamp,

        'simple copy from source column to destination column.'

);

INSERT INTO SYM\_TRANSFORM\_COLUMN

 (

        transform\_id,

        include\_on,

        target\_column\_name,

        source\_column\_name,

        pk,

        transform\_type,

        transform\_expression,

        transform\_order,

        last\_update\_time,

        last\_update\_by,

        create\_time,

        description

)

VALUES

(

        'Container\_InventoryChangeName',

        '\*',

        'handling\_qty',

        'handling\_qty',

        0,

        'copy',

        '',

        1,

        current\_timestamp,

        'Installation',

        current\_timestamp,

        'simple copy from source column to destination column.'

);

INSERT INTO SYM\_TRANSFORM\_COLUMN

 (

        transform\_id,

        include\_on,

        target\_column\_name,

        source\_column\_name,

        pk,

        transform\_type,

        transform\_expression,

        transform\_order,

        last\_update\_time,

        last\_update\_by,

        create\_time,

        description

)

VALUES

(

        'Container\_InventoryChangeName',

        '\*',

        'item\_code',

        '',

        0,

        'lookup',

        'SELECT item\_code from stock\_product WHERE stock\_product\_id = :stock\_product\_id',

        1,

        current\_timestamp,

        'Installation',

        current\_timestamp,

        'simple copy from source column to destination column.'

);

INSERT INTO SYM\_TRANSFORM\_COLUMN

 (

        transform\_id,

        include\_on,

        target\_column\_name,

        source\_column\_name,

        pk,

        transform\_type,

        transform\_expression,

        transform\_order,

        last\_update\_time,

        last\_update\_by,

        create\_time,

        description

)

VALUES

(

        'Container\_InventoryChangeName',

        '\*',

        'equipment\_zone\_code',

        '',

        0,

        'lookup',

        'SELECT equipment\_zone\_code from location WHERE warehouse\_code = :warehouse\_code AND location\_code = :location\_code',

        1,

        current\_timestamp,

        'Installation',

        current\_timestamp,

        'simple copy from source column to destination column.'

);

INSERT INTO SYM\_TRANSFORM\_COLUMN

 (

        transform\_id,

        include\_on,

        target\_column\_name,

        source\_column\_name,

        pk,

        transform\_type,

        transform\_expression,

        transform\_order,

        last\_update\_time,

        last\_update\_by,

        create\_time,

        description

)

VALUES

(

        'Container\_InventoryChangeName',

        '\*',

        'StockType',

        '',

        0,

        'lookup',

        'SELECT CASE WHEN Location\_code IN (''1BRICO'', ''OF'', ''KITOF'', ''BLISPACK'', ''BLISPACKO'', ''PINT'', ''PVNOVA'', ''PVNOVAO'') THEN ''OF'' WHEN Location\_code IN (''LOST'', ''UNDO'') THEN ''LOST'' WHEN Location\_code LIKE (''GER%'') THEN ''LOST'' WHEN Location\_code LIKE (''SGUR%'') THEN ''LOST'' WHEN Location\_code LIKE (''AGUARD%'') THEN ''WAIT'' WHEN Location\_code LIKE (''4CAIS%'') THEN ''WAIT'' WHEN Location\_code IN (''01W0'', ''01W1'', ''02W0'', ''02W1'', ''02W2'', ''04W0'', ''04W1'', ''6P01W0'', ''06P02W0'') THEN ''CAIS'' WHEN Location\_code IN (''04DOC01'', ''04DOC02'', ''04DOC03'', ''04DOC04'', ''04DOC05'', ''04DOCSOS'', ''01DOC01'', ''01DOC02'', ''01DOCNEI'', ''01DOCSAV'') THEN ''CAIS'' WHEN Location\_code = ''1REC01'' THEN ''REC'' WHEN Location\_code LIKE (''1CAI%'') THEN ''CAIS'' ELSE ''STOCK'' END AS StockType from location WHERE warehouse\_code = :warehouse\_code AND location\_code = :location\_code',

        1,

        current\_timestamp,

        'Installation',

        current\_timestamp,

        'simple copy from source column to destination column.'

);

