UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

Relatórios Técnicos
do Departamento de Informática Aplicada
da UNIRIO
nº 0012/2010

ANÁLISE DO USO DE MODELO FLEXÍVEL PARA AUTORIZAÇÃO DE ACESSO COM DATABASE LINK

Leonardo Guerreiro Azevedo
Léo Antunes

Departamento de Informática Aplicada

UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO
Av. Pasteur, 458, Urca - CEP 22290-240
RIO DE JANEIRO – BRASIL

Projeto de Pesquisa

Grupo de Pesquisa Participante

Patrocínio

PETROBRAS

Relatórios Técnicos do DIA/UNIRIO, No. 0012/2010
Editor: Prof. Sean W. M. Siqueira Setembro, 2010

Análise do Uso de Modelo Flexível para Autorização de Acesso com Database Link*

Leonardo Guerreiro Azevedo, Léo Antunes

Núcleo de Pesquisa e Prática em Tecnologia (NP2Tec)
Departamento de Informática Aplicada (DIA) – Universidade Federal do Estado do Rio de Janeiro (UNIRIO)

{azevedo, leo.antunes}@uniriotec.br

Abstract. Interest in researching on information security is growing in order to comply with commercial and government application security requirements. There are different methods for access authorization control. Azevedo *et al.* [2009, 2010] proposed a flexible framework for information authorization that can be used to controle database accesses. This work presents an evaluation of this proposal in distributed database considering Oracle technology.

Keywords: Flexible Model for Information Authorization, Access Autorization in distributed databases, database link.

Resumo. A pesquisa em segurança da informação tem recebido cada vez mais atenção a fim de atender às necessidades de segurança de aplicações comerciais e governamentais. Existem diferentes métodos para tratar controle de autorização de acesso. Azevedo *et al.* [2009, 2010] proporam um modelo flexível para autorização de informações que pode ser aplicada para controlar de acesso a banco de dados. Este trabalho apresenta a avaliação desta proposta em bancos de dados distribuídos considerando a tecnologia Oracle.

Palavras-chave: Modelo flexível para autorização de informações, Autorização de acesso em banco de dados distribuídos, database link.

* Trabalho patrocinado pela Petrobras.

Sumário

1 Introdução 7

2 Database link 7

2.1 Tipos de database link 8

2.2 Categorias de usuários para *database link* 9

2.3 Uso de SYS_CONTEXT com database links 13

2.4 Análise do uso de database link e modelo flexível 13

3 Análises práticas do uso de *Database Link* com o modelo flexível 15

3.1 Preparação do ambiente para os testes 15

3.2 Testes experimentais 17

3.2.1 Abordagem com utilização da função SYS_CONTEXT com *userenv* 18

3.2.2 Abordagem *Secure Session-based Application Context* 18

3.2.2.1 *Application Context* 19

3.2.2.2 Abordagem *Application Context Initialized Externally* 20

3.2.2.3 Abordagem *Application Context Initialized Globally* 20

3.2.2.4 Abordagem *Client Session-based Application Context* 21

3.2.3 Abordagem *Global Application Context* 21

4 Conclusões 24

5 Referências Bibliográficas 26

Figuras

Figura 1 – Exemplo de *database link* [Fogel *et al.*, 2010].....8
Figura 2 - Comando de criação de Database Link público 8
Figura 3 - Consulta à tabela lineitem no banco de dados remoto 15
Figura 4 - Predicado da tabela LINEITEM para a regra.....16
Figura 5 - Comando para aplicaçã da política T1 à tabela lineitem 16
Figura 6 - Criação do domínio de rede para criação do database link.....16
Figura 7 - Comando para criação de um database link privado com usuário conectado 17
Figura 8 - Comando para criação de um database link público com usuário conectado 17
Figura 9 - Comando para criação de um database link privado com usuário fixo 17
Figura 10 - Execução da consulta no banco de dados remoto pelo usuário Y2T0 17
Figura 11 - Execução da consulta no banco de dados remoto pelo usuário TPCH.....18
Figura 12 - Comando para criação do contexto 19
Figura 13 - Corpo do package 19
Figura 14 - Modificação da função genérica para captura do contexto do usuário 19
Figura 15 - Bloco de comando para incluir a chave do usuário no contexto 19
Figura 16 - Comando para criação do contexto ainf_ext.....20
Figura 17 - Comando para criação do contexto ainf_init_global.....20
Figura 18 - Armazenamento de valores de atributos de contexto global 22
Figura 19 - Comando para criação do contexto ainf_global 23
Figura 20 - Corpo do package ainf_ctx_global 23
Figura 21 - Código de recuperação da chave do usuário no contexto ainf_global 24
Figura 22 - Bloco de comando para incluir determinada chave de usuário identificada pela sessão no contexto.....24
Figura 23 - Bloco de comando para indicar ao banco o identificador do usuário.....24

Tabelas

Tabela 1 – Análise dos tipos de links....9

Tabela 2 - Exemplos de comandos SQL para criar *database link* 11

Tabela 3 – Descrição de como os usuários acessam o banco de dados remoto através do *link* 12

Tabela 4 – Tipos de contexto de aplicação 14

1 Introdução

Database Links permitem acessar dados e objetos de esquema em bancos de dados distribuídos [Fogel *et al.*, 2006]. Ao se definir políticas de segurança em banco de dados, deve-se tomar cuidado quando dados são acessados via *database link* [Jeloka *et al.*, 2008].

O objetivo deste trabalho é avaliar os problemas existentes e propor soluções em relação ao uso do modelo flexível [Azevedo *et al.*, 2009, 2010] implementado sobre o *Virtual Private Database* [Jeloka *et al.*, 2008] com *database link* considerando o SGBD (Sistema Gerenciador de Banco de Dados) Oracle.

Esse relatório está organizado em 5 capítulos, sendo o capítulo 1 a presente introdução. No capítulo 2, são apresentados os principais conceitos de *database link*. No capítulo 3, são apresentados resultados de análises práticas do uso de *database link* com o modelo flexível. Nos capítulos 4 e 5 são apresentadas nossas conclusões e as referências bibliográficas, respectivamente.

2 Database link

Esta seção apresenta os principais conceitos de *Database Link* de acordo com o manual "Oracle Database Administrator Guide" [Fogel *et al.*, 2006].

Um *database link* é um ponteiro que define um caminho de comunicação de modo único de um servidor de banco de dados Oracle para outro servidor de banco de dados. A Figura 1 ilustra um exemplo de *Database Link*. O ponteiro do *link* é na realidade definido como uma entrada de uma tabela do dicionário de dados. Para acessar o *link*, o usuário tem que estar conectado ao banco de dados local que contém a entrada do dicionário de dados.

Para que a conexão ocorra, cada banco de dados em um sistema distribuído deve ter um nome de banco de dados global único no domínio da rede. O nome de banco de dados global identifica unicamente o servidor de banco de dados no sistema distribuído.

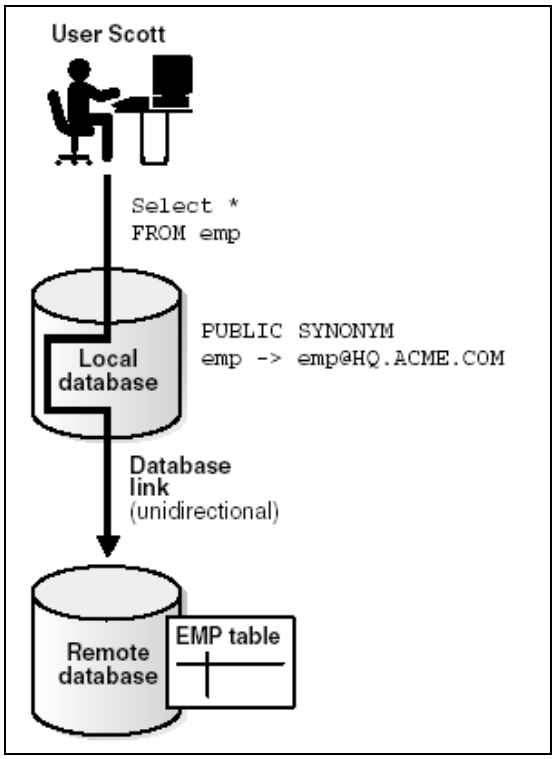

Figura 1 – Exemplo de *database link* [Fogel *et al.*, 2010]

Os privilégios para criar *database links* são: CREATE DATABASE LINK; CREATE PUBLIC DATABASE LINK; CREATE SESSION.

2.1 Tipos de database link

Existem três tipos de *database links*:

- *Public Database link*: é criado especificando a palavra chave *public* no comando de criação do *link*, como no exemplo apresentado na Figura 2.

```
CREATE PUBLIC DATABASE LINK foo USING 'sales';
```

Figura 2 - Comando de criação de Database Link público

  Todos os usuários e subprogramas PL/SQL no banco de dados podem usar o link para acessar os objetos de banco de dados do banco de dados remoto.

- *Global database link*: neste caso, um link de rede é criado. Quando uma rede Oracle usa um servidor de diretório, o servidor de diretório automaticamente cria e gerencia *global database links* (como nomes de serviços de rede) para cada banco de dados Oracle na rede. Usuários e subprogramas PL/SQL em qualquer banco de dados podem usar o *link* global para acessar os objetos no banco de dados remoto.

- *Private database link*: usuário e senha são utilizados para conectar ao banco, sendo que estas informações podem ser do usuário conectado (*connected user*), um usuário específico ou o *current user*, que não necessariamente é o usuário conectado. Os três comandos a seguir correspondem aos respectivos tipos de *database links*:

  ```
  CREATE DATABASE LINK supply.us.acme.com;
  ```

```
CREATE DATABASE LINK link_2 CONNECT TO jane IDENTIFIED
BY doe USING 'us_supply';

CREATE DATABASE LINK link_1 CONNECT TO CURRENT_USER
USING 'us_supply';
```

No primeiro comando foi utilizado o nome global do banco de dados para o banco de dados remoto *supply*. Nos outros comandos foi utilizado o nome do serviço *supply*. A Tabela 1 apresenta uma análise dos tipos de *links*.

Tabela 1 – Análise dos tipos de links

Tipo de Link	Características
Private database link	Tipo mais seguro. Apenas o proprietário do link ou os subprogramas dentro do mesmo esquema podem usar o link.
Public database link	Mais simples de usar quando muitos usuários precisam acessar um banco de dados remoto.
Global database link	Quando uma rede Oracle usa um servidor de diretório, um administrador pode gerenciar *global database links* para todos os bancos de dados no sistema. A gestão dos *database links* é centralizada e simples.

2.2 Categorias de usuários para *database link*

Ao criar um *database link*, são determinados quais usuários podem conectar ao banco de dados remoto para acesso aos dados. Existem as seguintes categorias de usuários: *connected user link*; *link* com usuário fixo; *current user*. Estas categorias são caracterizadas a seguir e analisadas em seguida em relação ao uso com o VPD.

- *Connected user links*

  Connected user links (*links* de usuário conectado) não têm *string* de conexão associadas a eles. A vantagem de um *link* de usuário conectado é que o usuário referenciando o *link* conecta ao banco de dados remoto com o mesmo usuário. Portanto, por não existir nenhuma *string* de conexão associada, nenhuma senha é armazenada em texto claro no dicionário do banco de dados.

  No entanto, *link* de usuário conectado tem algumas desvantagens. Este tipo de *link* requer que usuários tenham contas e privilégios no banco de dados remoto para o qual eles estão tentando se conectar. Logo, eles requerem mais administração de privilégios por parte de administradores de banco de dados. Além disso, atribuir mais privilégios que os usuários necessitam viola um conceito fundamental de segurança de privilégio mínimo: deve ser dado aos usuários apenas os privilégios que eles precisam para realizar suas tarefas.

  A capacidade de usar *database link* de usuário conectado depende de vários fatores. O mais importante entre eles é se o usuário é autenticado pelo banco de dados usando uma senha, ou autenticado externamente pelo sistema operacional ou por um serviço de autenticação da rede. Se o

usuário é autenticado externamente, então a habilidade para usar *link* de usuário conectado também depende se o banco de dados remoto aceita autenticação remota de usuários, a qual é ajustada pelo parâmetro REMOTE_OS_AUTHENT (true – um usuário autenticado externamente pode conectar ao banco de dados; false – caso contrário, ao menos que um protocolo seguro ou um serviço de autenticação na rede suportado pelo *Oracle Advanced Security* é usado).

- Link com usuário fixo

  No caso de *link* com usuário fixo, o usuário/senha definidos na criação *database link* são utilizados para todas as conexões ao banco de dados remoto. O usuário e a senha **não** são encriptados e são armazenados no dicionário de dados na tabela *LINK$*. Usuários podem conseguir obter estas informações (por exemplo, têm privilégio *SELECT ANY TABLE* e o parâmetro *07_DICTIONARY_ACCESSIBILITY* está ajustado para *TRUE*) e se conectarem no banco de dados remoto.

- Current user

  No caso do uso de um usuário global no *current user*, o usuário global deve ser autenticado por um certificado X.509 (um usuário da empresa autenticado via SSL) ou uma senha (usuário da empresa autenticado por senha), e ser um usuário em ambos os bancos de dados envolvidos no link. *Current user* faz parte da opção de segurança avançada no Oracle.

  O usuário invocando o *link CURRENT_USER* não precisa ser um usuário global. Por exemplo, se *jane* é autenticada (não como um usuário global) por senha para o banco de dados *Contas a Pagar*, ela pode acessar uma *stored procedure* para recuperar dados do banco de dados HQ. A *procedure* usa um *current user database link*, que a conecta a HQ como o usuário global *scott*. O usuário *scott* é um usuário global e é autenticado através de um certificado SSL, mas *jane* não é.

Algumas limitações importantes:

- Se o *current user database link* não está sendo acessado de dentro de um objeto armazenado, então o usuário corrente é o mesmo que o usuário conectado acessando o *link*. Por exemplo, se *scott* emite um *SELECT* através de um *current user database link*, então o usuário corrente considerado é *scott*.

- Ao executar um objeto armazenado, tais como *procedure, view* ou *trigger* que acessa o *link* do banco de dados, o usuário corrente é o usuário dono do objeto armazenado, e não o usuário que chama o objeto. Por exemplo, se *Jane* chama a procedure *scott.p* (criada por *scott*), e o *current user link* aparece dentro da *procedure* chamada, então *scott* é o *current user* do *link*.

- Não é permitido conectar a um banco de dados com usuário da empresa (*enterprise user*) e então usá-lo como *current user link* em uma *stored procedure* que existe em um esquema global, compartilhado. Por exemplo, se o usuário *Jane* acessa uma *stored procedure* no esquema compartilhado *guest* no banco de dados HQ, ela não pode usar *current user link* neste esquema para *logar* no banco de dados remoto.

Exemplos de comandos SQL para criar *database link* são apresentados na Tabela 2.

Tabela 2 - Exemplos de comandos SQL para criar *database link*

Comando SQL	Conecta ao banco de dados	Conecta como	Tipo de Link
CREATE DATABASE LINK sales.us.americas.acme_auto.com USING 'sales_us';	sales usando o serviço de nome de rede sales_us	Connected user	Private connected user
CREATE DATABASE LINK foo CONNECT TO CURRENT_USER USING 'am_sls';	sales usando o serviço de nome de rede am_sls	Current global user	Private current user
CREATE DATABASE LINK sales.us.americas.acme_auto.com CONNECT TO scott IDENTIFIED BY tiger USING 'sales_us';	sales usando o serviço de nome de rede sales_us	scott usando senha tiger	Private fixed user
CREATE PUBLIC DATABASE LINK sales CONNECT TO scott IDENTIFIED BY tiger USING 'rev';	sales usando o serviço de nome de rede rev	scott usando senha tiger	Public fixed user
CREATE SHARED PUBLIC DATABASE LINK sales.us.americas.acme_auto.com CONNECT TO scott IDENTIFIED BY tiger AUTHENTICATED BY anupam IDENTIFIED BY bhide USING 'sales';	sales usando o serviço de nome de rede sales	scott usando senha tiger, autenticado comoas anupam usando senha bhide	Shared public fixed user

Para consultar a tabela no banco de dados remoto:

```
SELECT * FROM emp@hq.acme.com;
```

```
SELECT * FROM emp@foo;
```

Sinônimos podem ser criados para esconder do usuário o nome do *link* do banco de dados. Um sinônimo permite acessar uma tabela no banco de dados remoto usando a mesma sintaxe que seria utilizada para acessar uma tabela no banco de dados local. Por exemplo, se fosse criado um sinônimo *emp* para *emp@hq.acme.com* então o seguinte comando SQL poderia ser emitido para obter os mesmos dados que o comando apresentado anteriormente.

```
SELECT * FROM emp;
```

Para acessar um objeto do esquema remoto, é necessário ter acesso ao objeto remoto no banco de dados. Portanto, para realizar qualquer *update*, *insert*¸ ou *delete* no objeto remoto, é necessário ter também o privilégio de *SELECT* no objeto, além dos privilégios de *update*, *insert*¸ ou *delete*. Diferentemente de quando se acessa um objeto local, o privilégio de *SELECT* é necessário para acessar o objeto remoto porque o banco de

dados não tem capacidade de descrição remota (*remote describe*). O banco de dados precisa executar um *SELECT* * no objeto remoto para determinar sua estrutura.

As seguintes operações não podem ser realizadas usando *database links*:

- Atribuir privilégios a objetos remotos;
- Executar operações *describe* em alguns objetos remotos. Sendo que é permitido executar estas operações em tabelas, visões, *procedures*, *packages* e funções;
- Analisar objetos remotos;
- Definir e garantir integridade referencial;
- Atribuir *roles* a usuários em banco de dados remoto;
- Obter *roles* não *default* no banco de dados remoto. Por exemplo, se *Jane* conecta no banco de dados local e executa uma *stored procedure* que usa um *link* de usuário fixo para conectar como *scott* no banco de dados remoto, *Jane* recebe os perfis *default* de *scott* no banco de dados remoto. *Jane* não pode emitir o comando *SET ROLE* para obter uma *role* não default.
- Executar consultas com junções resolvidas utilizando indexação *hash* que usam conexões compartilhadas;
- Usar um *current user link* sem autenticação através de SSL, senha, ou autenticação nativa NT.

Database links públicos ou privados podem ter autenticação ou não.

Tabela 3 – Descrição de como os usuários acessam o banco de dados remoto através do *link*

Tipo do Link	Autenticado	Acesso seguro
Privado	Não	O banco de dados remoto usa a informação de segurança da sessão local. Corresponde ao *connected user database link* Senha deve ser sincronizada entre os dois bancos de dados.
Privado	Sim	O usuário/senha são obtidos da definição do *link* ao invés de ser do contexto da sessão local. Corresponde ao *fixed user link.*
Público	Não	Funciona da mesma forma que o *link* privado sem autenticação, exceto que todos os usuários podem referenciar este ponteiro para o banco de dados remoto. Senha é armazenada em texto claro.
Público	Sim	Todos os usuários no banco de dados podem acessar o banco de dados remoto e todos usam o mesmo usuário/senha para realizar a conexão. Senha é armazenada em texto claro.

2.3 Uso de SYS_CONTEXT com database links

O contexto da aplicação local baseada em contexto pode ser acessado via comandos SQL dentro de uma sessão do usuário utilizando a função SQL *SYS_CONTEXT*. Quando estes comandos SQL envolvem *database links*, então a função SQL *SYS_CONTEXT* é executada no lado do *database link* que iniciou a conexão e captura o contexto presente neste site.

Se chamadas a procedimentos PL/SQL remotos são executadas sobre o *database link*, então qualquer função *SYS_CONTEXT* dentro destes procedimentos é executada no *database link* de destino. Neste caso, apenas contextos de aplicações iniciados externamente ficam disponíveis no *database link* de destino. Por razões de segurança, apenas as informações de contexto de aplicações inicializadas externamente são propagadas para o *database link* de destino a partir do site de inicialização do *database link*.

2.4 Análise do uso de database link e modelo flexível

No manual de segurança da Oracle (Oracle® Database Security Guide 10g Release 2 (10.2) [Jeloka *et al.*, 2008] é ressaltado que cuidados especiais devem ser tomados ao utilizar bancos de dados distribuídos e *database links*. No entanto, neste manual e em outros manuais (como em [Jeloka *et al.*, 2006]) não são apresentadas soluções ou análises para tratar o problema do uso de *Virtual Private Database* com *database links*.

A partir dos estudos a respeito de database link apresentados nas seções anteriores, podemos dividir o problema em dois casos:

i. Os usuários da empresa corresponderem a usuários de banco de dados e serem utilizados para fazer a conexão no banco local e remoto;

ii. Os usuários da empresa não corresponderem a usuários de banco de dados. Neste caso, um mesmo usuário de banco de dados é utilizado para realizar as operações no banco de dados local e remoto, e as informações do usuário final ficam cadastradas em tabelas de perfil gerenciadas em um modelo à parte, como o modelo de perfil × usuário proposto por [Azevedo *et al.*, 2009, 2010].

Para tratar o caso (i), a melhor proposta seria a escolha de uma das categorias de usuários para *database links*. Analisando estas categorias observamos que:

- *link* com usuário fixo: não pode ser utilizado com VPD, pois todo o acesso ao banco de dados remoto é feito pelo o usuário dono do link e não pelo usuário que está emitindo a consulta.
- *current user*: depende do uso de um certificado e pode-se fazer um mapeamento entre o usuário local e o usuário global que ele utiliza, os quais podem ser diferentes. Nestes dois casos, têm-se mais trabalho de gestão de certificado e de mapeamento de usuários locais e globais.

  No entanto, o uso do *current user* traz problema quando utilizado com *procedure, view* ou *trigger*, pois o usuário que utilizado nestes casos é sempre o usuário dono destes objetos e não o usuário que os está executando.

  Além disso, não é permitido conectar a um banco de dados com usuário da empresa e então usá-lo como *current user* link em uma *stored procedure* que existe em um esquema global, compartilhado.

- *connected user link*: A vantagem de um link de usuário conectado é que o usuário referenciando o *link* conecta ao banco de dados remoto com o mesmo usuário. No entanto, este tipo de *link* requer que usuários tenham contas e privilégios no banco de dados remoto para o qual eles estão tentando se conectar.

Portanto, para o caso (i) dos usuários das aplicações estarem cadastrados como usuários do banco de dados, e estes usuários serem utilizados para fazer os acessos através de um *database link*, o uso de *connected user link* mostra-se como mais apropriado.

Para o caso (ii), o uso de categorias de usuários de *database link* não se configura como solução, pois o mesmo usuário de banco de dados é utilizado para todos os usuários finais. Logo, a melhor solução para este caso é o uso de *SYS_CONTEXT* com *database links*. De acordo com o que foi apresentado na seção 1.3, só ocorrerá um problema no uso de *SYS_CONTEXT* quando a função *SYS_CONTEXT* for utilizada dentro de uma *procedure* ou *function* invocada no banco de dados remoto.

Contextos de aplicação baseado em sessão podem ser inicializados externamente ou globalmente. Em ambos os casos, as informações de contexto ficam armazenadas na sessão do usuário [Jeloka *et al.*, 2008].

- Inicialização externa pode vir de uma interface OCI, um processo em fila de trabalho (*job queue*), ou de um *connected user database link*;
- Inicialização global usa atributos e valores de uma localização centralizada, tal como um diretório LDAP.

A inicialização do contexto de aplicação segura externamente permite especificar um tipo especial de *namespace* que aceita a inicialização de valores de atributos de recursos externos e os armazena em sessão de usuário local. Permitir que o contexto de aplicação segura seja inicializado externamente garante desempenho e permite a propagação automática de atributos de uma sessão para outra. *Connect user database links* são suportados apenas por contextos de aplicações inicializadas de fontes externas baseadas em OCI. A Tabela 4 apresenta um resumo dos diferentes tipos de contexto de aplicação.

Tabela 4 – Tipos de contexto de aplicação

Tipo de Contexto de Aplicação	Armazenado na UGA	Armazenado na SGA	Suporta Database Link com usuário conectado	Suporta armazenamento centralizado do contexto de aplicação de usuários	Suporta aplicações multicamada sem sessão
Contexto de Aplicação	SIM	NÃO	NÃO	NÃO	NÃO
Contexto de Aplicação Inicializado Externamente	SIM	NÃO	SIM	NÃO	NÃO
Contexto de Aplicação Inicializado Globalmente	SIM	NÃO	NÃO	SIM	NÃO
CLIENT CONTEXT	SIM	NÃO	SIM	NÃO	SIM

Contexto de Aplicação Global	NÃO	SIM	NÃO	NÃO	SIM

É importante ressaltar que comandos que referenciam *database links*, clusters, índices ou sinônimos não podem ser auditados diretamente. Entretanto, acessos podem ser indiretamente auditados pela auditoria das operações que afetam a tabela base.

3 Análises práticas do uso de *Database Link* com o modelo flexível

Nesta seção são apresentados testes de uso de database link considerando políticas armazenadas segundo o modelo flexível apresentado em Azevedo *et al.* [2009, 2010]. Os testes experimentais foram realizados de acordo com consultas e dados do Benchmark TPC-H [TPC-H 2009]. As abordagens avaliadas foram: uso da função SYS_CONTEXT com *userenv*, e as abordagens *Secure Session-Based Application Context* (de gerência de contexto armazenado na *User Global Area* (UGA)) e *Global Application Context* (de gerência de contexto armazenado na *System Global Area* (SGA)) [Jeloka *et al.*, 2008].

3.1 Preparação do ambiente para os testes

Inicialmente, o ambiente foi configurado no banco de dados remoto:

- O modelo de dados do *benchmark* TPC-H foi criado no banco de dados remoto e os dados foram carregados nas respectivas tabelas;
- O modelo flexível foi criado no banco de dados remoto [Azevedo *et al.*, 2009, 2010];
- Usuários de teste foram criados no banco de dados local e remoto com mesmo nome e senha.
- A consulta a ser executada e qual regra a ser aplicada foram definidas;
- A regra foi incluída no modelo flexível do banco de dados remoto;
- A função genérica foi aplicada à tabela sobre a qual a regra foi definida;
- As informações do banco de dados remoto foram configuradas no *tnsnames* banco de dados local;
- Os *database links* foram criados no banco de dados local;
- Consulta foi executada no banco de dados local.

A seguir são detalhadas algumas características desta configuração.

- Consulta de teste: A consulta executada nos testes é apresentada na Figura 3. Observe que ela foi escrita utilizando o *database link dl.ainf* cuja forma de criação é descrita para cada avaliação de tipo de *database link*.

```
SELECT count(*) from tpch.lineitem@dl.ainf;
```

Figura 3 - Consulta à tabela lineitem no banco de dados remoto

- Regra de autorização de acesso: Para os testes foi considerada a regra T1, apresentada em Azeve *et al.* [2009] ajustada para considerar a restrição sobre o balança na conta. A regra é apresentada a seguir e foi aplicada à tabela LINEITEM. O predicado retornado é apresentado na Figura 4.

T1. Regra: Restringir o acesso pelo Gerente de vendas do hemisfério Norte das regiões Ásia e América aos itens de pedidos provenientes de fornecedores das nações desta área que possuem balanço na conta superior a R$9.000,00.

Perfil: Gerente de vendas de baixo risco Ásia e América, com acesso às informações relativas aos fornecedores com balanço na conta superior a R$9.000,00 do hemisfério Norte das regiões Ásia e América.

```
  1=2 OR (l_suppkey in (
    select s_suppkey
    from supplier, nation, region
    where s_nationkey = n_nationkey
    and n_regionkey = r_regionkey
    and n_hemisphere IN (1)
    AND r_regionkey IN (1 , 2)
    AND s_acctbal >= 9000))
```

Figura 4 - Predicado da tabela LINEITEM para a regra

- Aplicação da política: Em seguida, aplicamos a política no banco de dados remoto com o usuário de sistema (*System*) com o comando apresentado na Figura 5.

```
EXECUTE dbms_rls.add_policy(object_schema => 'TPCH', object_name =>
'LINEITEM', policy_name => 'POLITICA_T1', function_schema => 'PERFIL',
policy_function => 'F_POLICY', update_check => true);
```

Figura 5 - Comando para aplicaçã da política T1 à tabela lineitem

- Configuração do banco de dados remoto: Em relação aos tipos de database link criados, é importante ressaltar que antes de criá-los devemos incluir as informações do banco de dados remoto no arquivo *tnsnames* localizado na pasta "C:\oracle\product\10.2.0\<server_name>\network\ADMIN". O comando para inclusão desta informação é apresentado na Figura 6.

```
  ORCL =
    (DESCRIPTION =
      (ADDRESS = (PROTOCOL = TCP)(HOST = [host_name])(PORT =
port_number))
      (CONNECT_DATA =
        (SERVER = DEDICATED)
        (SERVICE_NAME = [service_name])
      )
    )
```

Figura 6 - Criação do domínio de rede para criação do database link

- Os comados para criação dos database links, privado com usuário conectado, público com usuário conectado e privado com usuário fixo, foram:
    - Privado com usuário conectado

      Para criar um *database link* privado com usuário conectado, devemos abrir uma conexão com o usuário para o qual queremos criar o *database link* e executar o comando apresentado na Figura 7.

```
CREATE DATABASE LINK dl.ainf using 'DLAINF';
```

Figura 7 - Comando para criação de um database link privado com usuário conectado

- Público com usuário conectado

  Para criar um *database link* público com usuário conectado, devemos abrir uma conexão com o usuário de sistema (*System*) e executar o comando apresentado na Figura 8.

```
CREATE PUBLIC DATABASE LINK pdl.ainf using 'DLAINF';
```

Figura 8 - Comando para criação de um database link público com usuário conectado

- Privado com usuário fixo

  Para criar um *database link* privado com usuário fixo, devemos abrir uma conexão com o usuário para o qual queremos criar o *database link* e executar o comando apresentado na Figura 9.

```
CREATE DATABASE LINK fdl.ainf CONNECT TO <username> IDENTIFIED BY
<password> USING 'DLAINF';
```

Figura 9 - Comando para criação de um database link privado com usuário fixo

3.2 Testes experimentais

Nesta seção são apresentados detalhes dos testes realizados para o uso de *database link* utilizando o modelo flexível no ambiente configurado de acordo com o que foi apresentado na seção 3.1 e de acordo com as abordagens: uso da função SYS_CONTEXT com o *namespace userenv*, *Secure Session-based Application Context* e *Global Application Context*.

Os usuários utilizados para realizar os testes foram Y2T0 e TPCH. O usuário Y2T0 possui o perfil Gerente de vendas de baixo risco Ásia e América, o qual restringe o seu acesso de acordo com o predicado apresentado na Figura 4. O usuário TPCH, tem o privilégio EXEMPT, POLICY, ou seja, nenhuma regra de autorização é aplicada sobre ele.

Quando a consulta apresentada na Figura 3 foi executada com o usuário Y2T0 diretamente no banco de dados onde está o *dabase link*, o resultado obtido foi que nenhum registro foi retornado (Figura 10).

```
COUNT(*)
----------
0
```

Figura 10 - Execução da consulta no banco de dados remoto pelo usuário Y2T0

Executando a mesma consulta com o usuário TPCH, o resultado obtido para todos os testes foi que 6.001.204 registros foram retornados (Figura 11), ou seja, todas os registros da tabela.

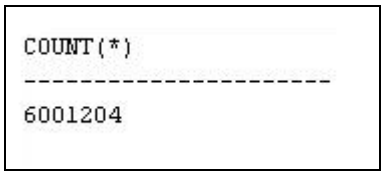

Figura 11 - Execução da consulta no banco de dados remoto pelo usuário TPCH

3.2.1 Abordagem com utilização da função SYS_CONTEXT com *userenv*

Userenv é um *namespace* de contexto de aplicação que o servidor de banco de dados Oracle provê. Este *namespace* disponibiliza atributos pré-definidos. Por exemplo, o nome de usuário, endereço IP do usuário conectado, ou um nome de usuário do *proxy* se a conexão do usuário é feita através de uma camada intermediária. Estes atributos são palavras primitivas de sessão, que são informações que o banco de dados captura em relação a um usuário de sessão [Jeloka *et al.*, 2008].

Utilizando esta abordagem, para todos os tipos de *database link*, a política foi aplicada corretamente. Ou seja, executando a consulta apresentada na Figura 3 com usuário Y2T0 no banco de dados local (acessando o banco de dados remoto), o resultado obtido foi a quantidade de registros que o usuário tem acesso (147.319). Enquanto que executando a consulta com o usuário TPCH, o resultado obtido foram todos os registros da tabela (6.001.204). No entanto, esta abordagem apresenta as seguintes limitações:

- Solução não serve para arquiteturas de múltiplas camadas.
- Aplicação não pode ajustar atributos do *namespace userenv*.
- Oracle recupera informações do usuário como usuário conectado ou usuário do sistema operacional.

3.2.2 Abordagem *Secure Session-based Application Context*

A *Secure Session-based Application Context* é uma abordagem onde os valores dos atributos do contexto ficam armazenados na *User Global Area* (UGA), que é uma área da memória que armazena informações relacionadas à sessão.

Session-based application contexts podem ser inicializados por fontes externas ou podem ser inicializados globalmente. Em ambos os casos, a informação do contexto é armazenada na sessão do usuário. Os contextos de aplicação baseados em sessão que são inicializados externamente podem aceitar inicializações de atributos e valores por fontes externas assim como uma interface OCI, um *job queue process*, ou um *database link* com usuário conectado. Já os contextos que são inicializados globalmente, eles podem aceitar inicializações de atributos e valores de um local centralizado, tal como um diretório LDAP [Jeloka *et al.*, 2008].

Nossos testes consistem na análise destes três casos: *application context*, *application context initialized externally* e *application context initialized globally*.

Para criar um *session-based application context*, é necessário associá-lo a um *package*, pois somente as *procedures* deste *package* poderão definir valores para os atributos desse contexto (através da função *SET_CONTEXT*). Como todos os casos são baseados na mesma abordagem, todos os packages têm a mesma estrutura. Ou seja, eles possuem apenas uma *procedure* chamada *set_chave_usuario* a qual define na sessão o atributo identificador do usuário. Essa definição é feita utilizando a função *SET_CONTEXT*, que recebe como parâmetros o nome do contexto, o nome do atributo e o valor do atributo.

O nome do atributo é "chave_usuario" e o valor do atributo é o parâmetro passado para a *procedure* "set_chave_usuario".

O comando para criação do contexto é apresentado na Figura 12, onde as palavras-chave INITIALIZE EXTERNALLY e INITIALIZE GLOBALLY são opcionais, para obtermos um comportamento específico**Erro! Fonte de referência não encontrada.**. A estrutura do corpo dos *packages* é apresentada na Figura 13.

```
CREATE CONTEXT <nome_contexto> USING <nome_package> [INITIALIZE
EXTERNALLY | INITIALIZE GLOBALLY];
```

Figura 12 - Comando para criação do contexto

```
CREATE OR REPLACE
PACKAGE BODY <nome_package> AS
    PROCEURE set_chave_usuario (chave_usuario varcha2) AS
    BEGIN
      DBMS_SESSION.SET_CONTEXT('<nome_contexto>',
                               'chave_usuario',chave_usuario);
    END set_chave_usuario;
END ainf_ctx;
```

Figura 13 - Corpo do package

Os contextos e seus packages associados foram criados no banco de dados local e no banco de dados remoto.

Em seguida, também em todos os casos, a função genérica do modelo flexível 'F_POLICY' foi modificada no banco de dados remoto para identificar o usuário local utilizando o contexto criado, como apresentado na Figura 14**Erro! Fonte de referência não encontrada.**.

```
v_user := lower(sys_context('<nome_contexto>', 'chave_usuario'));
```

Figura 14 - Modificação da função genérica para captura do contexto do usuário

Antes de realizar a consulta de acesso ao banco de dados remoto, a chave do usuário deve ser incluída na sessão utilizando a função do *package* do banco de dados local. Este ajuste é feito através da invocação da *procedure set_chave_usuario*, a partir da conexão do banco de dados local, como demonstrado na Figura 15Para este caso, realizamos.

```
Begin
    perfil.<nome_package>.set_chave_usuario('<user>');
end;
```

Figura 15 - Bloco de comando para incluir a chave do usuário no contexto

3.2.2.1 *Application Context*

Para este caso, realizamos a consulta apresentada na Figura 3 nos três tipos de *database links*: *database link* privado com usuário conectado, *database link* público com usuário conectado e *database link* privado com usuário fixo. Para os três casos, com o usuário Y2T0 atribuído ao contextoPara este caso, realizamos, o resultado obtido foi que nenhum registro foi retornado. Enquanto que com o usuário TPCH, todos os registros

da tabela foram retornados (6.001.204). Portanto, o uso de *Application Context* com os tipos de *database link* testados não funciona como desejado.

3.2.2.2 Abordagem *Application Context Initialized Externally*

Initialized Externally é um recurso do Oracle que permite que você especifique um tipo especial de *namespace* que aceite inicializações de valores de atributos de um recurso externo e armazene-os na sessão local do usuário. *Externally* indica que o *namespace* pode ser inicializado utilizando uma interface OCI quando uma sessão é estabelecida. Isto aumenta o desempenho e permite a propagação automática de atributos de uma sessão para outra.

Somente os contextos de aplicação inicializados por fontes externas baseadas na OCI suportam database links com usuário conectado [Jeloka *et al.*, 2008].

Neste caso, na criação do contexto, devemos especificar a palavra chave INITIALIZED EXTERNALLY, como apresentado na Figura 16.

```
CREATE CONTEXT <nome_contexto> USING <nome_package> INITIALIZED
EXTERNALLY;
```

Figura 16 - Comando para criação do contexto ainf_ext

Em seguida, realizamos a consulta apresentada na Figura 3 nos três tipos de *database links*: *database link* privado com usuário conectado, *database link* público com usuário conectado e *database link* privado com usuário fixo. Para os três casos, o resultado obtido foi a quantidade de registros que o usuário tem acesso (147.319). Então, podemos deduzir que o contexto foi propagado. Quando realizamos a consulta com o usuário TPCH, todos os registros da tabela foram retornados (6.001.204).

3.2.2.3 Abordagem *Application Context Initialized Globally*

Esta funcionalidade utiliza um local centralizado para armazenar o contexto de aplicação do usuário, possibilitando que aplicações inicializem o contexto de aplicação durante a inicialização, baseada na identidade do usuário. Em particular, ela tem suporte a *labels* e privilégios do *Oracle Label Security*. Esta funcionalidade torna mais fácil para o administrador a gestão de contextos para um número grande de usuários e bancos de dados. Por exemplo, muitas organizações querem administrar informações de usuário centralmente, em um diretório baseado em LDAP.

Session-based application context initialized globally utiliza o *Lightweight Directory Access Protocol* (LDAP). LDAP é um protocolo de acesso a diretórios, padrão, extensível, e eficiente. O diretório LDAP armazena uma lista de usuários que os quais estão atribuídos à aplicações. Um servidor de banco de dados Oracle pode usar o *Oracle Internet Directory*, ou diretórios de terceiros como o *Microsoft Active Directory* e *Sun Microsystems iPlanet*, como serviços de diretórios para autenticação de usuários coorporativos.

Neste caso, na criação do contexto devemos especificar a palavra-chave INITIALIZED GLOBALLY, como apresentado na Figura 17.

```
CREATE CONTEXT ainf_init_global USING ainf_ctx_init_global INITIALIZED
GLOBALLY;
```

Figura 17 - Comando para criação do contexto ainf_init_global

Mais uma vez, realizamos a consulta apresentada na Figura 3 nos três tipos de *database links*: *database link* privado com usuário conectado, *database link* público com

usuário conectado e *database link* privado com usuário fixo. Então, para os três casos, com o usuário Y2T0 atribuído ao contexto, o resultado obtido foi que nenhum registro foi retornado. Por outro lado, quando realizamos a consulta com o usuário TPCH, todos os registros da tabela foram retornados (6.001.204). No entanto, vale ressaltar que, para uma melhor avaliação do uso deste tipo de contexto, é necessário criar um LDAP e considerá-lo nos testes.

3.2.2.4 Abordagem *Client Session-based Application Context*

Na abordagem *Client Session-based Application Context*, o desenvolvedor implementa código na aplicação para utilizar funções da *Oracle Call Interface* (OCI) para ajustar e apagar informações de sessão do usuário, a qual é então armazenada na *User Global Area* (UGA).

A vantagem deste tipo de *application context* é que uma aplicação pode verificar se existem dados específicos da sessão do usuário, ao invés do banco de dados ter que realizar esta tarefa. Outra vantagem é que as chamadas para ajustar o valor do contexto são incluídas na próxima chamada ao servidor, que melhora o desempenho.

Porém, a segurança de um *application context* está de acordo com um *client session-based application context*: qualquer usuário da aplicação pode ajustar o *client application context*, e nenhuma verificação é realizada no banco de dados.

Você configura o *client session-based application context* somente para a aplicação cliente. Você não configura nada no servidor de banco de dados no qual o cliente se conecta. Qualquer ajuste do contexto de aplicação no servidor de banco de dados não afeta o *client session-based application context*.

Para configurar um client session-based application context, utilize a função da OCI OCIAppCtxSet. Um *client session-based application context* utilize o namespace CLIENTCONTEXT, atualizável por um cliente OCI ou por um package existente da DBMS_SESSION para contexto de aplicação.

O *namespace* CLIENTCONTEXT permite que uma única transação de uma aplicação mude a informação de contexto do usuário e use o mesmo tratamento de sessão do usuário para servir a nova requisição do próprio usuário [HUEY *et al*., 2007]

3.2.3 Abordagem *Global Application Context*

A *Global Application Context* é uma abordagem onde os valores dos atributos do contexto ficam armazenados na *System Global Area* (SGA), que é uma área da memória definida para cada instância[1] do Oracle. Isso significa que os atributos do contexto são definidos para toda a instância, ou seja, se em uma sessão o usuário do banco define o valor de um atributo, todas as suas outras sessões concorrentes verão esse mesmo valor [Jeloka *et al*., 2008].

Outra característica do contexto global é que os valores dos atributos são associados a identificadores, de forma que o mesmo atributo pode receber mais de um valor para

[1] Cada bacon de dados Oracle executando está associado com uma instância do Oracle. Quando uma instância é iniciada no servidor do banco de dados (independente do tipo de computador), o Oracle aloca uma área de memória chamada *Sytem Global Area* (SGA) e inicia o processo Oracle. Esta combinação da SGA com um processo Oracle é chamada de instância. A memória e o processo de uma instância gerencia eficientemente os dados do banco de dados assicados e serve a um ou mais usuários do banco de dados.

cada identificador definido. Os valores podem ser depois recuperados através deste identificador.

Nesse caso, é necessário estipular o usuário do banco para o qual o atributo é definido (usuário do banco utilizado pela aplicação), pois como o valor do atributo será armazenado em uma área de memória global, devemos explicitar que somente esse usuário terá acesso a tal valor do atributo. Aliados aos identificadores, isso permite que cada usuário possua vários valores para cada atributo, de modo que o identificador único dos valores será a dupla "usuário × identificador".

A Figura 18 ilustra o armazenamento de valores de atributos em um contexto global. Nesse exemplo, existe um contexto global onde foram criados dois atributos, chamados "ATR1" e "ATR2" respectivamente. Além disso, tanto para o usuário "DBUSER1" quanto para o usuário "DBUSER2", foram definidos dois valores para cada atributo.

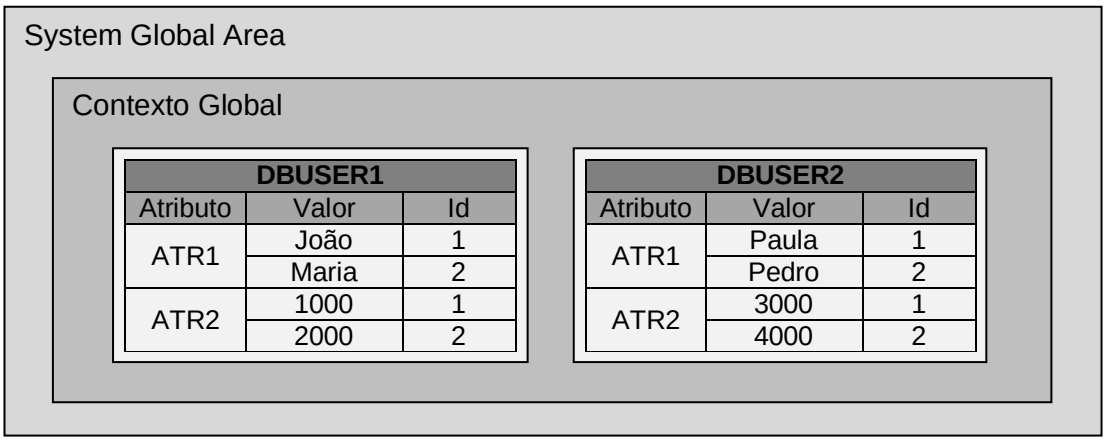

Figura 18 - Armazenamento de valores de atributos de contexto global

Note que, apesar de existirem identificadores repetidos, a dupla "usuário × identificador" permite acessar um único valor para cada atributo. Por exemplo, se o usuário "DBUSER1" tentar acessar o valor do "ATR1" com o identificador "1" ativado, ele receberá o valor "João".

A vantagem do uso da SGA em relação à UGA é que na SGA o contexto do usuário pode ser definido apenas uma vez e ser reutilizado por todas as sessões do usuário. Na UGA, para cada conexão que o usuário realiza (ou seja, para cada abertura de sessão), o contexto deve ser definido novamente.

A SGA utiliza o atributo CLIENT_IDENTIFIER do namespase *userven*. Este atributo é ajustado utilizando a interface DBMS_SESSION, para associar a sessão do banco de dados com um usuário ou grupo específico.

A interface DBMS_SESSION para gestão de contexto de aplicações tem um identificador de cliente para cada contexto de aplicação. Dessa forma, o contexto da aplicação pode ser gerenciado globalmente. A seguinte interface no DBMS_SESSION permite o administrador gerenciar contexto de aplicação em sessões cliente:

- SET_CONTEXT
- CLEAR_CONTEXT
- CLEAR_ALL_CONTEXT
- SET_IDENTIFIER
- CLEAR IDENTIFIER

O servidor de aplicação pode usar SET_CONTEXT para ajustar o contexto de aplicação para o ID de um cliente específico. Então, quando atribuindo uma conexão de banco de dados para processar a requisição do cliente, o servidor de aplicação necessita emitir um SET_IDENTIFIER para especificar o ID a sessão da aplicação. A partir daí, toda vez que o cliente invocar SYS_CONTEXT, apenas o contexto que está associado com o identificar é retornado.

Se a aplicação gerar um ID de sessão para usar como um CLIENT_IDENTIFIER, então o id da sessão deve ser randômico e protegido através da rede por criptografia. Se o ID da sessão não é randômico, então usuários maliciosos podem adivinhar o ID da sessão e acessar os dados de outro usuário. Se ID da sessão não é criptografado através da rede, então um usuário malicioso pode recuperar o ID da sessão e acessar a conexão.

Da mesma forma que a abordagem anterior, para criar um contexto global, é necessário associá-lo a um *package*, pois somente as *procedures* deste *package* poderão definir valores para os atributos desse contexto (através da função *SET_CONTEXT*). Além disso, no caso desta abordagem, para que esta funcione corretamente, o contexto deve ser criado tanto no banco de dados remoto quanto no banco de dados local.

Então, criamos o contexto *ainf_global* associado ao *package ainf_ctx_global*, através do comando apresentado na Figura 19. O *package ainf_ctx_global* é apresentado na Figura 20. Devemos criar o contexto na máquina local e na máquina remota.

```
CREATE CONTEXT ainf_global USING ainf_ctx_global ACCESSED GLOBALLY;
```

Figura 19 - Comando para criação do contexto ainf_global

```
create or replace
PACKAGE BODY ainf_ctx_global AS
  PROCEDURE set_chave_usuario (id_sessao_usuario number,
                               chave_usuario varchar2) AS
  v_usuario_banco varchar2 (100);
  BEGIN
    v_usuario_banco := lower(sys_context('userenv', 'session_user'));
    if v_usuario_banco != 'secuser' then
      null;
    else
      v_usuario_banco := lower(sys_context('userenv','proxy_user'));
    end if;
    DBMS_SESSION.SET_CONTEXT('ainf_global', 'chave_usuario',
                             chave_usuario, v_usuario_banco,
                             id_sessao_usuario);
  END set_chave_usuario;
END ainf_ctx_global;
```

Figura 20 - Corpo do package ainf_ctx_global

Em seguida, a função genérica do modelo flexível 'F_POLICY' foi modificada no banco de dados remoto para identificar o usuário local utilizando o contexto criado 'ainf_global', como apresentado na Figura 21.

```
v_user := lower(sys_context('ainf_global', 'chave_usuario'));
```

Figura 21 - Código de recuperação da chave do usuário no contexto ainf_global

Empregando esta abordagem, antes de realizar a consulta de acesso ao banco de dados remoto, faremos algo parecido com o que foi feito na abordagem anterior. A diferença nesse caso é que devemos gerar, a partir da conexão do banco de dados local, um identificador (correspondente ao parâmetro *idSessaoUsuario*) para definir o contexto e informar a chave do usuário para armazenar na sessão identificada. Isto é, invocando a *procedure set_chave_usuario*, como apresentado na Figura 22, a partir do *package* do banco de dados local. Em seguida, antes de executar a consulta, devemos indicar ao banco o identificador do usuário, o que é feito com o comando apresentado na Figura 23.

```
Begin
    perfil.ainf_ctx.set_chave_usuario(<idSessaoUsuario>, '<user>');
end;
```

Figura 22 - Bloco de comando para incluir determinada chave de usuário identificada pela sessão no contexto

```
  Begin
    DBMS_SESSION.SET_IDENTIFIER(<idSessaoUsuario>);
  end;
```

Figura 23 - Bloco de comando para indicar ao banco o identificador do usuário

A consulta foi então realizada nos três tipos de *database links*: *database link* privado com usuário conectado, *database link* público com usuário conectado e *database link* privado com usuário fixo. Para os três casos, executamos a consulta da Figura 3 com o usuário Y2T0 atribuído ao contexto em conjunto com o identificador *idSessaoUsuario* com valor igual a 1 (comandos apresentados na Figura 22 e na Figura 23). O resultado obtido foi a quantidade de registros que o usuário tem acesso (147.319), ou seja, podemos deduzir que o contexto foi propagado. Enquanto que com o usuário TPCH, todos os registros da tabela foram retornados (6.001.204).

4 Conclusões

Database Links permitem acessar dados e objetos de esquema em bancos de dados distribuídos [Fogel *et al.*, 2006]. Um *database link* é um ponteiro que define um caminho de comunicação de modo único de um servidor de banco de dados Oracle para outro servidor de banco de dados. Jeloka *et al.* [2008] ressaltam que ao se definir políticas de segurança em banco de dados, deve-se tomar cuidado quando dados são acessados via *database link*.

Os tipos de *database link* foram apresentados (público, privado e global). As categorias de usuários foram descritas (usuário conectado, usuário fixo e usuário atual). Comandos para criação de cada tipo de link com as diferentes categorias de usuários foram apresentados e suas limitações foram discutidas.

Em seguida, foram realizadas análises práticas do uso de *database link* com o modelo flexível (FARBAC - Flexible Approach for Role-Based Access Control) [Azevedo *et al*, 2009, 2010]. Os testes experimentais incluíram avaliações das abordagens para tratamento do contexto do usuário (ajuste e recuperação) a fim de aplicar as funções de autorização de informações da maneira devida. As seguintes abordagens foram avaliadas:

- Utilização da função SYS_CONTEXT com *namespace userenv*: A aplicação das políticas ocorreu corretamente, no entanto, esta abordagem apresenta as seguintes limitações:
    - Solução não serve para arquiteturas de múltiplas camadas.
    - Aplicação não pode ajustar atributos do *namespace userenv*.
    - Oracle recupera informações do usuário como usuário conectado ou usuário do sistema operacional.
- *Session-based Application Context* (UGA): nesta abordagem os atributos de contexto são armazenados na User Global Area (UGA), sendo a sessão do usuário armazenada durante uma conexão com o banco. Além de serem armazenados na UGA diretamente, também existem as possibilidades dos contextos poderem ser obtidos de forma global (quando depende da existência de um diretório LDAP criado), de forma externa (onde o contexto é armazenado no banco de dados remoto) ou serem ajustados pela própria aplicação utilizando *Oracle Call Interface* (OCI). Estes três tipos e os resultados com os testes foram:
    - *Application Context*: nenhuma política foi aplicada corretamente.
    - *Application Context Initialized Externally*: políticas foram aplicadas corretamente, mas há a necessidade do *package* que define o contexto e os usuários estarem replicados no banco de dados local e remoto.
    - *Application Context Initialized Globally:* políticas não foram aplicadas corretamente, mas acredita-se que isto ocorreu porque não havia um diretório LDAP criado. Logo, é necessário refazer os testes considerando a implementação do mesmo. Nesta abordagem, é necessário que o *package* que define o contexto e os usuários estarem replicados no banco de dados local e remoto.
    - *Client session-based application context*: esta abordagem utiliza o *namespace* CLIENTCONTEXT, o qual atualizável por uma aplicação cliente utilizando funções OCI ou por um package existente da DBMS_SESSION para contexto de aplicação. Este caso tem uma limitação séria de segurança, pois qualquer aplicação cliente pode acessar o contexto. O teste experimental não foi realizado para este caso pelos seguintes motivos:
        - Limitação de segurança.
        - Requer intervenções no código das aplicações para ajustar contexto.
- *Global Application Context*: Nesta abordagem os valores dos atributos do contexto ficam armazenados na *System Global Area* (SGA), que é uma área da memória definida para cada instância do Oracle. Os atributos de contexto são válidos para todas as conexões que acessem a referida instância do Oracle. A aplicação das políticas, neste caso, ocorreu corretamente.

Portanto, o contexto não é propagado de forma a poder ser utilizado o modelo flexível para as abordagens *Session-based Application Context* e *Session-based Application Context Initialized Globally*. Sendo que a última necessita ser testada com o diretório LDAP. O uso do *namespace userenv* apesar de funcionar, não pode ser utilizado devido

às limitações apresentadas. As soluções possíveis de uso são *Session-based Application Context Initialized Externally* e *Global Application Context*.

5 Referências Bibliográficas

AZEVEDO, L.; DUARTE, D.; PUNTAR, S.; ROMEIRO, C.; BAIAO, F.; CAPPELLI, C. **Avaliação Prática de Funcionalidades para Autorização de Informações (Label Security e Virtual Private Database).** Relatórios Técnicos do DIA/UNIRIO (RelaTe-DIA), RT-0026/2009, 2009. Disponível em <http://seer.unirio.br/index.php/monografiasppgi/article/view/543/557>. Acesso em 24 Ago. 2010.

AZEVEDO, L. G., PUNTAR, S., THIAGO, R., Baião, F., Cappelli, C. **A Flexible Framework for Applying Data Access Authorization Business Rules.** In: 12th International Conference on Enterprise Information Systems (ICEIS 2010), Funchal, Madeira, Portugal, June 8-12, 2010.

FOGEL, S., LANE, P., AUSTIN, D., *et al*. **Oracle® Administrator's Guide 10g Release 2.** Oracle Corporation, 2006. Disponível em <http://download.oracle.com/docs/cd/B19306_01/server.102/b14231.pdf>. Acesso em 24 Mar. 2010.

HUEY, P., JELOKA, S., GOSSELIN, D., KUCHEROV, S., LEWIS, N. *et al.* **Oracle Database Security Guide, Oracle RDBMS 11Gr1**. Oracle Corporation, 2007. Disponível em <http://www.comp.dit.ie/btierney/oracle11gdoc/network.111/b28531.pdf>. Acesso em 27 Julho. 2010.

JELOKA, S., MULAGUND, G., LEWIS, N. *et al*. **Oracle Database Security Guide, Oracle RDBMS 10gR2**. Oracle Corporation, 2008. Disponível em <http://download.oracle.com/docs/cd/B19306_01/network.102/b14266.pdf>. Acesso em 7 Jan. 2009.

TPCH, **TPC Benchmark H**. Transaction Processing Performance Council. Disponível em http://www.tpc.org/tpch/. Acesso em: 10 set. 2009.