ORACLE VIRTUAL PRIVATE DATABASE

Vesna Lukšić
Hrvatska elektroprivreda d.d.
Sektor za informatiku i telekomunikacije

Oracle VPD

- **Virtual Private Database** (VPD)
- **Fine-grained access control**: veže sigurnosnu politiku na objekte baze podataka
- VPD administratorima omogućuje definiranje i provođenje kontrole pristupa (prema aplikacijskim ili sesijskim postavkama) na razini retka - **Row Level Security (RLS).**

Zašto VPD?

- **Skalabilnost** – Tabela „Zaposlenici” sadrži 15.000 zapisa. Pretpostavimo da želimo omogućiti zaposlenicima pristup vlastitim podacima. Korištenjem view-a, trebalo bi se kreirati 15.000 view-a. Korištenjem VPD, zahtjev se rješava jednom policy funkcijom.

- **Jednostavnost** – Pretpostavimo da je na tabelu T kreirano mnogo view-a. Pretpostavimo da želimo ograničiti pristup nekim informacijama u tablici. Bez primjene VPD, definicije svih view-ova bi se trebale promijeniti. Korištenjem VPD-a, dovoljno je kreirati policy funkciju na tablici jer se ona primjenjuje i u svim view-ima kreiranim nad tablicom T.

- **Fleksibilnost** – Mogu se definirati različite sigurnosne politike za SELECT, INSERT, UPDATE, DELETE ili INDEX naredbe.

- **Sigurnost** – Dodavanjem sigurnosne politike na tablice, view-ove ili sinonime, VPD osigurava jednaku kontrolu pristupa bez obzira kako korisnik pristupa podacima.

Oracle VPD

Objekt ID	Naziv objekta	Org. jedinica	Mjesto
2797	TS 10/0,4 kV	4002	011495
3745	TS 10/0,4 kV “Šafran”	4004	040835
3717	TS 10/0,4 kV"Štefanec I"	4004	064025
16475	TS 10(20)/0,4 kV SELINE	4012	056995

Korisnik 1 – može vidjeti i ažurirati podatke org. jedinice 4002

Korisnik 2 – može vidjeti i ažurirati podatke org. jedinice 4004

Oracle VPD

- Kako VPD radi?

  Kada korisnik pristupi tablici (ili view-u ili sinonimu) koja je „zaštićena” VPD policy-em (funkcijom),

  1. Oracle poslužitelj poziva policy funkciju.
  2. Policy funkcija vraća predikat koji je baziran na postavkama sesije ili na sadržaju baze podataka.
  3. Poslužitelj dinamički mijenja dobiveni upit dodajući predikat u WHERE klauzulu.
  4. Izvodi se tako modificirani SQL upit.

Implementacija VPD-a

DBMS_RLS

Vlasnik SCHEME

Korisnik 2

Koristi DBMS_RLS paket za postavljanje policy-a na tablicu OBJEKTI

VPD policy automatski dodaje u WHERE klauzulu org_jed = 4004

SELECT * from OBJEKTI

SELECT * from OBJEKTI WHERE org_jed = 4004

Objekt ID	Naziv objekta	Org. jedinica	Mjesto
2797	TS 10/0,4 kV	4002	011495
3745	TS 10/0,4 kV “Šafran”	4004	040835
3717	TS 10/0,4 kV"Štefanec I"	4004	064025
16475	TS 10(20)/0,4 kV SELINE	4012	056995

Policy funkcija

Implementacija VPD-a

Koraci u implementaciji VPD-a:

- Definirati kontrolu pristupa
  - poslovna pravila
  - kontrolne tablice
- Kreirati policy funkciju
- Definirati policy
- Testirati postavke

Application Context

- Application context se koristi za definiranje korisničkih postavki kao što su:
  - ime aplikacije koja se koristi
  - korisničko ime
  - ostale sigurnosne postavke

- Application context se postavlja pozivom package-a dbms_session.set_context.

- Application context može se postaviti:
  - prilikom prijave korisnika pomoću database trigger-a
  - bilo kada pozivom ***dbms_session package-a***

Application Context

- Postoji Oracle-ov predefinirani application context – *"USERENV".*

ACTION, AUDITED_CURSORID, AUTHENTICATED_IDENTITY, AUTHENTICATION_DATA, AUTHENTICATION_METHOD, BG_JOB_ID, CLIENT_IDENTIFIER, CLIENT_INFO, CURRENT_BIND, CURRENT_SCHEMA, CURRENT_SCHEMAID, CURRENT_SQL, CURRENT_SQLn, CURRENT_SQL_LENGTH, DB_DOMAIN, DB_NAME, DB_UNIQUE_NAME, ENTRYID, ENTERPRISE_IDENTITY, FG_JOB_ID, GLOBAL_CONTEXT_MEMORY, GLOBAL_UID, HOST, IDENTIFICATION_TYPE, INSTANCE, INSTANCE_NAME, IP_ADDRESS, ISDBA, LANG, LANGUAGE, MODULE, NETWORK_PROTOCOL, NLS_CALENDAR, NLS_CURRENCY, NLS_DATE_FORMAT, NLS_DATE_LANGUAGE, NLS_SORT, NLS_TERRITORY, OS_USER, POLICY_INVOKER, PROXY_ENTERPRISE_IDENTITY, PROXY_GLOBAL_UID, PROXY_USER, PROXY_USERID, SERVER_HOST, SERVICE_NAME, SESSION_USER, SESSION_USERID, SESSIONID, SID, STATEMENTID, TERMINAL

- Select SYS_CONTEXT ('USERENV', 'CURRENT_USER') from dual;

Application Context

- Moguće je kreirati vlastiti application context i njegove atribute.

- Za postavljanje vrijednosti atributa koristi se DBMS_SESSION package:

  DBMS_SESSION.SET_CONTEXT (‘ime_env’, ‘ime_atributa’, vrijednost);

- Za čitanje vrijednosti atributa koristi se SYS_CONTEXT:

  SYS_CONTEXT (‘ime_env’, ‘ime_atributa’);

Application Context: kreiranje

1. Kreira se PL/SQL package koji postavlja kontekst:

```
CREATE OR REPLACE package body TERRA.context_package is
procedure set_context is
  the_user varchar2(30);
begin
   dbms_session.set_context('TERRA','SETUP','TRUE');
   the_user := sys_context('USERENV', 'SESSION_USER');
   set_organization(the_user);
   dbms_session.set_context('TERRA','SETUP','FALSE');
end set_context;
procedure set_organization (the_user varchar2) is
.....
 dbms_session.set_context('TERRA','ORG_JEDINICA', the_organization);
.....
.....
end context_package;
```

Application Context: kreiranje

2. Kreira se context i poveže sa odgovarajućim package-om:

```
CREATE OR REPLACE CONTEXT TERRA
 USING TERRA.CONTEXT_PACKAGE;
```

Svi atributi u „TERRA” context-u moraju biti postavljeni u terra.context_package-u.

Application Context: upotreba

3. Prilikom logiranja potrebno je pokrenuti postavljanje context-a:

```
CREATE OR REPLACE TRIGGER TERRA.terra_set_context
AFTER LOGON ON DATABASE
BEGIN
    TERRA.context_package.set_context;
END terra_set_context;
```

Context je moguće pokrenuti i iz aplikacije kojom se pristupa na bazu.

Application Context: upotreba

4. Kreira se VPD funkcija:

```
CREATE OR REPLACE package body TERRA.security_package is
function upisnik_select_security (Owner varchar2, Objname varchar2)
return varchar2 is
Predicate varchar2(2000);
begin
   Predicate := '1=2';
   if sys_context('USERENV','SESSION_USER') = 'TERRA' then
     Predicate := null;
   else
     Predicate :=
     'substr(ho_org_jed,1,length(sys_context(''TERRA'',''ORG_JEDINICA''))) =
      sys_context(''TERRA'',''ORG_JEDINICA'')';
   end if;
   return Predicate;
end upisnik_select_security;
```

Policy: kreiranje

Policy se definira korištenjem DBMS_RLS package-a:

```
BEGIN
  DBMS_RLS.ADD_POLICY    (
   object_schema       => 'TERRA'
   ,object_name        => 'HEP_OBJEKTI'
   ,policy_name        => 'UPISNIK_INSERT_POLICY'
   ,function_schema     => 'TERRA'
   ,policy_function     =>'SECURITY_PACKAGE.UPISNIK_INSERT_SECURITY'
   ,statement_types     => 'INSERT'
   ,policy_type        => dbms_rls.dynamic
   ,long_predicate      => FALSE
   ,update_check        => TRUE
   ,static_policy      => FALSE
   ,enable             => TRUE );
END;
```

Definiranje Policy-a

- DROP_POLICY (***DBMS_RLS.DROP_POLICY*** (

```
object_schema   => 'TERRA'
,object_name    => 'HEP_OBJEKTI'
,policy_name    => 'UPISNIK_INSERT_POLICY');
```

- REFRESH_POLICY (***DBMS_RLS.DROP_POLICY*** (

```
object_schema   => 'TERRA'
,object_name    => 'HEP_OBJEKTI'
,policy_name    => 'UPISNIK_INSERT_POLICY');
```

- ENABLE_POLICY (***DBMS_RLS.ENABLE_POLICY*** (

```
object_schema   => 'TERRA'
,object_name    => 'HEP_OBJEKTI'
,policy_name    => 'UPISNIK_INSERT_POLICY‘
,enable    => TRUE);
```

Policy: kreiranje

- Moguće je definirati više policy-a na jedan objekt baze podataka.
  - Policy-e se povezuje s AND operatorom.

Policy: kreiranje

Moguće je kreirati VPD policy funkcije i bez korištenja application context-a.
Umjesto toga, mogu se postavljati upiti u bazu iz policy funkcija.

```
function kamate_k_security (Owner varchar2, Objname varchar2)
return varchar2 is
Predicate varchar2(2000);
begin
    Predicate := '1=2';
    if sys_context('USERENV','SESSION_USER') = 'USURAE' then
      Predicate := null;
    else
      Predicate := 'K_USER = sys_context(''USERENV'',''SESSION_USER'')';
   end if;
   return Predicate;
end kamate_k_security;
```

Korisnici mogu pristupiti samo onim recima u kojima je zapisano njihovo korisničko ime.

Policy: kreiranje

```
BEGIN
 SYS.DBMS_RLS.ADD_POLICY    (
  object_schema        => 'USURAE'
  ,object_name         => 'KAMATNE_STOPE_USER'
  ,policy_name         => 'KAMATE_K_POLICY'
  ,function_schema     => 'USURAE'
  ,policy_function     =>
   'USURAE_SECURITY_PACKAGE.KAMATE_K_SECURITY'
  ,statement_types     => 'SELECT,INSERT,UPDATE,DELETE,INDEX'
  ,policy_type         => dbms_rls.dynamic
  ,long_predicate      => FALSE
  ,sec_relevant_cols   => 'K_USER'
  ,sec_relevant_cols_opt => NULL
  ,update_check        => FALSE
  ,static_policy       => FALSE
  ,enable              => TRUE );
END;
```

Tipovi policy-a

- **Dynamic –** defaultna vrijednost, policy funkcija se izvodi prilikom svakog SQL upita
- **Context_sensitive -** Policy funkcija se izvodi u samo kada je detektirana promjena application context-a.
- **Shared_context_sensitive –** Vrijede ista pravila kao i za CONTEXT_SENSITIVE s tim da se ova vrsta policy-a može dijeliti između više objekata koji koriste istu policy funkciju.
- **Static -** Policy funkcija se izvodi jednom i njen se rezultat (predikat) sprema u Shared Global Area (SGA).
- **Shared_static –** Dopušta se dijeljenje predikata između više objekata koji koriste istu policy funkciju.

Grupe policy-a

Moguće je grupirati više policy-a i vezati ih za neku aplikaciju.

U tom slučaju se definira application context koji određuje koja grupa policy-a se aktivira. Stoga kada korisnik pristupa nekom objektu u bazi, baza konzultira context kako bi odredila koja je grupa policy-a na snazi i aktivira sve policy-e iz te grupe.

Grupe policy-a su korisne u situacijama kada više aplikacija pristupa istom objektu u bazi podataka. Ovisno o tome iz koje se aplikacije pristupa, aktiviraju se odgovarajući policy-e.

Procedure za grupe policy-a

- DBMS_RLS procedure:
  - ADD_GROUPED_POLICY
  - ADD_POLICY_CONTEXT
  - CREATE_POLICY_GROUP
  - DELETE_POLICY_GROUP
  - DISABLE_GROUPED_POLICY
  - DROP_GROUPED_POLICY
  - DROP_POLICY_CONTEXT
  - ENABLE_GROUPED_POLICY
  - REFRESH_GROUPED_POLICY

Column-level VPD

- Umjesto postavljanja na tablicu ili view, policy se može postaviti na određene kolone u tablici ili view-u (nije moguće kod sinonima)
  - Default: query vraća ograničen broj redaka – samo one čije vrijednost u koloni je dopušteno vidjeti.
  - Mask: query vraća sve retke s NULL vrijednostima za kolone koje nije dopušteno vidjeti

- Ograničenja
  - Primjenjivo samo na SELECT naredbe
  - Predikat mora biti jednostavan uvjetni (Booleov) izraz

Column-level VPD

```
SYS.DBMS_RLS.ADD_POLICY   (
   object_schema       => 'USURAE'
   ,object_name        => 'KAMATNE_STOPE_USER'
   ,policy_name        => 'KAMATE_K_POLICY'
   ,function_schema    => 'USURAE'
   ,policy_function    => 'USURAE_SECURITY_PACKAGE.KAMATE_K_SECURITY'
   ,statement_types    => 'SELECT,INSERT,UPDATE,DELETE,INDEX'
   ,policy_type        => dbms_rls.dynamic
   ,long_predicate     => FALSE
   ,sec_relevant_cols  => 'K_USER'
   ,sec_relevant_cols_opt => dbms_rls.ALL_ROWS
   ,update_check       => FALSE
   ,static_policy      => FALSE
   ,enable             => TRUE );
```

U aplikacijama koje se bave izračunima ili koje ne očekuju NULL vrijednosti, bolje je koristiti standardni column-level VPD.

VPD privilegije

- Za kreiranje VPD policy-a potrebne su slijedeće privilegije u bazi:
  - EXECUTE on DBMS_RLS
  - Package uključuje add_policy, drop_policy, enable_policy, ...
  - CREATE PROCEDURE za kreiranje policy funkcije (nije potrebno ako se koristi postojeća policy funkcija).
- Za kreiranje application context-a:
  - CREATE ANY CONTEXT
  - CREATE PROCEDURE
  - EXECUTE on DBMS_SESSION
  - Odgovarajuće privilegije na objektima koji se koriste

Izuzeci

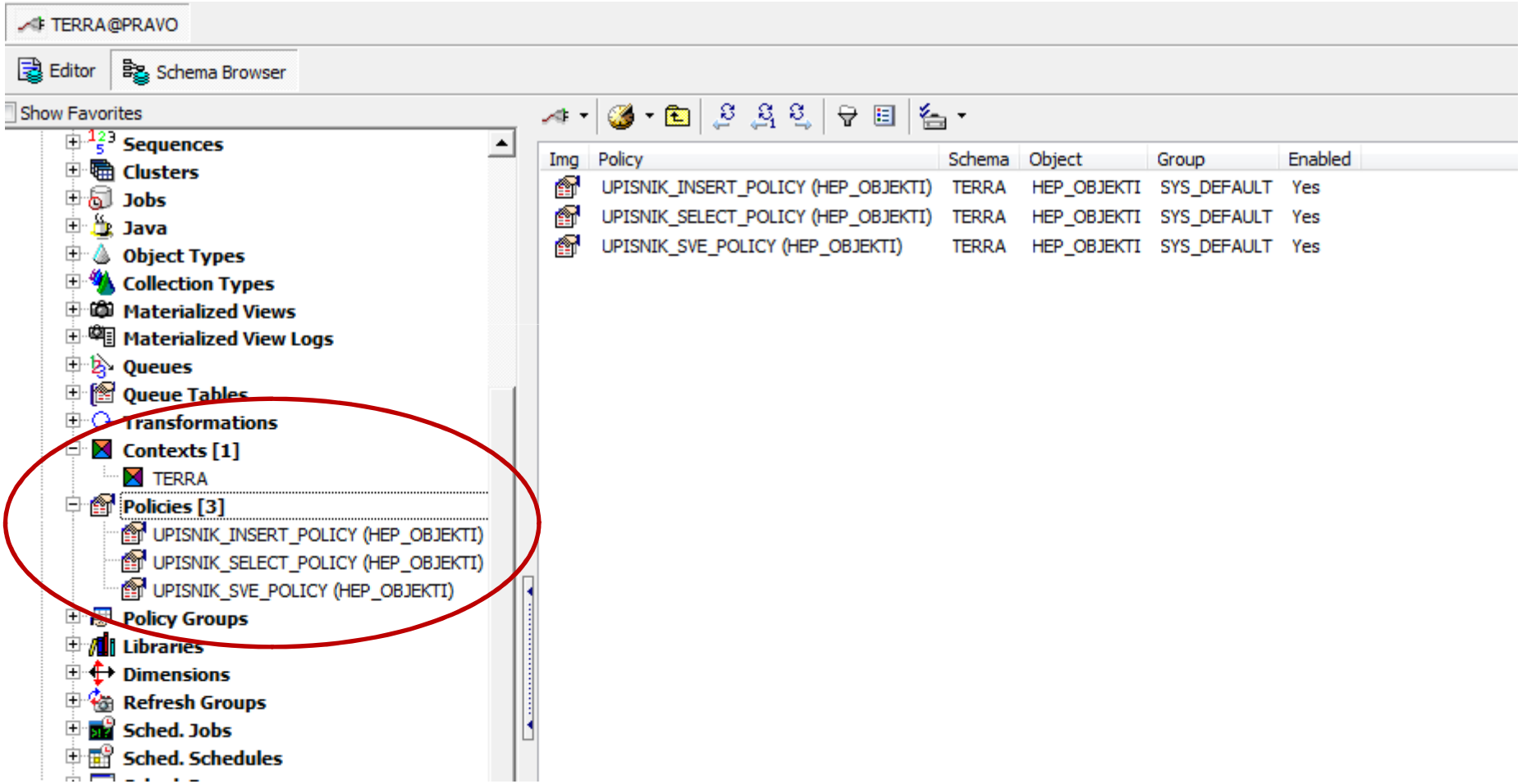
- VPD se ne provodi prilikom DIRECT path export-a
- VPD policy se ne može primijeniti na objektima u SYS schemi
- Svi database korisnici s EXEMPT ACCESS POLICY privilegijom (direktno dodijeljene ili indirektno – kroz rolu) su izuzeti od primjene VPD policy-a. Stoga dodjela te povlastice treba biti čvrsto kontrolirana u uvjetima provedbe VPD.
- Administratori mogu pomoću VPD policy-a ograničiti kreiranja indeksa specificiranjem INDEX naredbe u “statement_types” parameter. Kreiranje indeksa nije dozvoljeno ako korisnik nema puni pristup objektu baze.

Data dictionary

View	Opis
DBA_POLICES	Sadrži popis svih policy-a u bazi podataka
ALL_POLICIES	Sadrži popis svih policy-a korisnika
USER_POLICIES	Sadrži popis svih policy-a korisnika
V$VPD_POLICY	Lista svih policy-a – izvedenih i korištenih
ALL_CONTEXT	Lista svih context-a koje korisnik ima ili ih može vidjeti
SESSION_CONTEXT	Lista svih aktivnih context-a u trenutnoj sesiji

Toad

Toad DBA Suite for Oracle



Zaključak

- VPD je snažan alat za kontrolu pristupa bazi podataka.
- Korisnici ne mogu zaobići sigurnosne politike ugrađene u aplikacije, jer se sigurnosna politika vezuje na podatke.
- Teško je, ako ne i nemoguće, provjeriti da li ili ne određeni korisnik ima pristup određenom podatku u određenoj tablici u određenim okolnostima.
  Takva provjera zahtijevala bi provjeru svih policy funkcija.
  Stoga bi, prema definiranim poslovnim pravilima, trebalo pažljivo definirati policy funkcije.

Hvala na pažnji!

vesna.luksic@hep.hr