@cczarski folgen

ORACLE

Security-Basics
Privilegien, Rollen, SQL und PL/SQL - inkl. 12*c*-Update

Carsten Czarski, ORACLE Deutschland B.V. Co. KG

Themen

- Rechte, Rollen und PL/SQL: Grundsätzliches
  - Invokers vs. Definers Privileges
  - Wann nutzt man was?
  - Secure Application Roles
  - Was ist neu in Oracle12*c?*
  - Wie sieht das eigentlich bei Views aus ...?
- Security abseits von Privilegien und Rollen
  - Virtual Private Database (VPD)
  - Ressourcenschutz für Netzwerk und Dateisystem

Rechte, Rollen und PL/SQL

Wer bin ich und was darf ich?


```
create or replace package body plsql_xxx is
  procedure whoami is begin
    dbms_output.put_line('USER:            '||user);
    dbms_output.put_line('CURRENT_SCHEMA:  '||sys_context('userenv','CURRENT_SCHEMA'));
    dbms_output.put_line('CURRENT_USER:    '||sys_context('userenv','CURRENT_USER'));
  end whoami;

  procedure showroles is begin
    for i in (select role from session_roles) loop
      dbms_output.put_line(i.role);
    end loop;
  end showroles;

  procedure showprivs is begin
    for i in (select privilege from session_privs) loop
      dbms_output.put_line(i.privilege);
    end loop;
  end showprivs;

  procedure exec_sql(p_sql in varchar2) is begin
    execute immediate p_sql;
  end exec_sql;
end plsql_xxx;
```

Rechte, Rollen und PL/SQL
Wer bin ich und was darf ich?

- PL/SQL Code wird zwei Mal erzeugt
  - AUTHID DEFINER (Default)
  - AUTHID CURRENT_USER
  - In der Package Spec angeben

```
create or replace package plsql_inv authid current_user
is
  procedure whoami;
  procedure showroles;
  procedure showprivs;
  procedure exec_sql(p_sql in varchar2);
end plsql_inv;
/
sho err
```

Danach: Test als Eigentümer

AUTHID DEFINER

```
SQL> exec plsql_def.whoami
USER:              OWN
CURRENT_SCHEMA:    OWN
CURRENT_USER:      OWN
```

AUTHID CURRENT_USER

```
SQL> exec plsql_inv.whoami
USER:              OWN
CURRENT_SCHEMA:    OWN
CURRENT_USER:      OWN
```

- USER:
  User, mit dem der physikalische Connect stattfindet
- CURRENT_USER
  User, dessen Privilegien gerade aktiv sind
- CURRENT_SCHEMA
  Ersetzt das Schema-Präfix (keine Auswirkung auf Privilegien)

Danach: Test als anderer Nutzer
EXECUTE Privilegien sind eingeräumt

AUTHID DEFINER

```
SQL> exec own.plsql_def.whoami
USER:            CAL
CURRENT_SCHEMA:  OWN
CURRENT_USER:    OWN
```

AUTHID CURRENT_USER

```
SQL> exec own.plsql_inv.whoami
USER:            CAL
CURRENT_SCHEMA:  CAL
CURRENT_USER:    CAL
```

- USER:
  User, mit dem der physikalische Connect stattfindet
- CURRENT_USER
  User, dessen Privilegien gerade aktiv sind
- CURRENT_SCHEMA
  Ersetzt das Schema-Präfix (keine Auswirkung auf Privilegien)

Welche Privilegien sind aktiv?
Test als Eigentümer

AUTHID DEFINER

```
SQL> exec plsql_def.showprivs

ALTER SESSION
UNLIMITED TABLESPACE
CREATE VIEW
```

AUTHID CURRENT_USER

```
SQL> exec plsql_inv.showprivs

CREATE SESSION
ALTER SESSION
UNLIMITED TABLESPACE
CREATE TABLE
CREATE CLUSTER
CREATE VIEW
CREATE SEQUENCE
CREATE PROCEDURE
CREATE TRIGGER
CREATE TYPE
CREATE OPERATOR
CREATE INDEXTYPE
SET CONTAINER
```

Welche Privilegien habe ich per Rolle?
Test als Eigentümer

AUTHID DEFINER

```
SQL> exec plsql_def.showroles
```

AUTHID CURRENT_USER

```
SQL> exec plsql_inv.showroles

CONNECT
RESOURCE
```

Rollen sind in PL/SQL-Objekten mit AUTHID DEFINER abgeschaltet. Der PL/SQL-Code läuft mit den Rechten des Eigentümers, aber *ohne Rollen.*

Welche Privilegien sind aktiv?
Test als anderer User

AUTHID DEFINER

```
SQL> exec plsql_def.showprivs

ALTER SESSION
UNLIMITED TABLESPACE
CREATE VIEW
```

Privilegien des Eigentümers

AUTHID CURRENT_USER

```
SQL> exec own.plsql_inv.showprivs

CREATE SESSION
SET CONTAINER
```

Privilegien des Aufrufers

Zusammenfassung

- PL/SQL Code mit Definers' Privileges
  - AUTHID DEFINER – Default
  - Rollen sind abgeschaltet
  - Sinnvoll, wenn die Rechte des Eigentümers *benötigt werden* – bspw. für DML auf Tabellen im Schema

- PL/SQL-Code mit Invokers' Privileges
  - AUTHID CURRENT_USER
  - Rollen sind aktiv
  - Sinnvoll unter anderem, wenn ...
    ... PL/SQL in einem hoch privilegierten Schema ...
    ... von anderen Nutzern mit wenig Rechten ...
    ... zur Arbeit in *deren Schema* / auf *deren Tabellen* ...
    ... aufgerufen werden soll.

Secure Application Roles

- Wer den "geheimen Code" kennt, bekommt die Rolle ...

```
CREATE ROLE app_role
IDENTIFIED USING own.proc_grant_role;

create or replace procedure own.grantrole(
  p_code in number
) authid current_user is
begin
  if p_code = 42 then
    execute immediate 'set role app_role';
  else
    raise_application_error(-20000, 'FALSC
  end if;
end grantrole;

grant execute on own.grantrole to public;
```

Die Rolle wird *nicht* per GRANT an den User vergeben.

Allein die PL/SQL-Prozedur aktiviert die Rolle.

Rollen für PL/SQL-Objekte

ORACLE DATABASE 12c

- Oracle12*c* erlaubt den GRANT einer Rolle direkt an ein PL/SQL Objekt

```
grant {role} to package|procedure|function {plsqlobject}
```

  - Rolle ist dann *nur* im PL/SQL-Objekt aktiv
  - Für AUTHID DEFINER und CURRENT_USER
  - Damit Rollen in Definers' Rights Procedures!

```
SQL> Grant RESOURCE to package PLSQL_DEF;

SQL> exec own.plsql_def.showroles
RESOURCE
```

Neu: INHERIT PRIVILEGES

ORACLE DATABASE 12c

- Relevant bei AUTHID CURRENT_USER
- Ein Beispiel ...
  - User SCOTT ("normale" Privilegien) programmiert ...

```
create or replace package app_pkg authid current_user is
  procedure useful_dba_utility;
end app_pkg;

create or replace package body app_pkg is
  procedure useful_dba_utility is
  begin
    execute immediate 'grant dba to SCOTT';
    dbms_output.put_line('Temporary application tables purged.');
  end useful_dba_utility;
end app_pkg;
```

Oracle12*c*: INHERIT PRIVILEGES

ORACLE DATABASE 12c

- Relevant bei AUTHID CURRENT_USER
  - PL/SQL-Code mit Rechten des aufrufenden Users
  - Was ist, wenn ein DBA die Prozedur aufruft?
  - Auf einmal hat der Code *zu viele* Rechte
- Oracle12*c*: Recht "INHERIT PRIVILEGES"
  - Damit der PL/SQL-Code die Rechte des aufrufenden Users annehmen *kann*, ist dieses Recht nötig
  - Bei Erzeugen eines Users werden INHERIT Privilegien an PUBLIC vergeben → Verhalten wie in 11g
  - Mit Ausnahme von SYS

INHERIT PRIVILEGES

ORACLE DATABASE 12c

- Aufruf eines IR Package durch SYS

```
SYS@[sccloud027:1521/pdb01]
SQL> exec scott.app_pkg.useful_dba_utility
BEGIN scott.app_pkg.useful_dba_utility; END;

*
FEHLER in Zeile 1:
ORA-06598: Nicht ausreichende INHERIT PRIVILEGES-Berechtigung
ORA-06512: in "SCOTT.APP_PKG", Zeile 2
ORA-06512: in Zeile 1
```

INHERIT PRIVILEGES

ORACLE DATABASE 12c

- Aufruf eines IR Package durch CAL

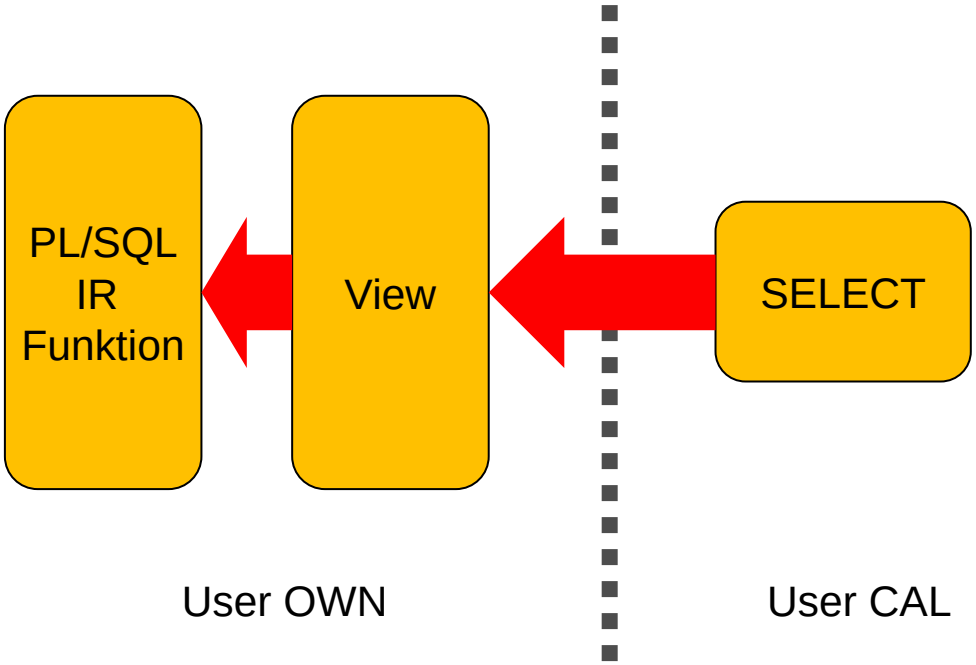
```
CAL@[sccloud027:1521/pdb01]
SQL> exec own.plsql_inv.whoami
USER:            CAL
CURRENT_SCHEMA:  CAL
CURRENT_USER:    CAL

PL/SQL-Prozedur erfolgreich abgeschlossen.

SYS@[sccloud027:1521/pdb01]
SQL> revoke inherit privileges on user cal from public;

Benutzerzugriff wurde aufgehoben (Revoke).

CAL@[sccloud027:1521/pdb01]
SQL> exec own.plsql_inv.whoami
BEGIN own.plsql_inv.whoami; END;

*
FEHLER in Zeile 1:
ORA-06598: Nicht ausreichende INHERIT PRIVIL
ORA-06512: in "OWN.PLSQL_INV", Zeile 2
ORA-06512: in Zeile 1
```

Beim Erzeugen des Users CAL wurde INHERIT PRIVILEGES ON USER CAL an PUBLIC vergeben. Somit kann PL/SQL-Code von OWN mit den Rechten von CAL ablaufen

Nach dem REVOKE kann PL/SQL-Code von OWN nicht mehr die Rechte von CAL nutzen.

Oracle12*c*: Privilegien und Views

- Eine IR-Funktion wird von einer View verwendet
- Die View wird von einem anderen User selektiert
- Mit wessen Rechten läuft die IR-Funktion ab ...?



Oracle12*c*: Privilegien und Views

- Beispielcode: Die Objekte gehören OWN

```
create or replace function own.func_whoami
return varchar2 authid current_user as
begin
  return sys_context('userenv','current_user');
end;

create or replace view own.view_whoami as
select func_whoami from dual;

grant select on own.view_whoami to cal;
```

```
CAL@[sccloud027:1521/pdb01]
SQL> select * from own.view_whoami ;

FUNC_WHOAMI
------------------------------------------
OWN

1 Zeile wurde ausgewählt.
```

Was gibt die View zurück, wenn CAL sie selektiert?

BEQUEATH CURRENT_USER

ORACLE DATABASE 12c

- Beispielcode

```
create or replace function func_whoami
return varchar2 authid current_user as
begin
  return sys_context('userenv','current_user');
end;

create or replace view view_whoami bequeath current_user as
select func_whoami from dual;

grant select on view_whoami to cal;
```

```
CAL@[sccloud027:1521/pdb01]
SQL> select * from own.view_whoami ;

FUNC_WHOAMI
------------------------------------------
CAL

1 Zeile wurde ausgewählt.
```

Vermischtes
Wussten Sie schon, dass ...

- INSERT-Rechte auf einzelne Spalten möglich sind?
- UPDATE natürlich auch!
- Andere Spalten werden auf NULL gesetzt

```
SCOTT@[sccloud033:1521/orcl]
Grant insert (EMPNO, ENAME) on SCOTT.EMP to INSERTER;

INSERTER@[sccloud033:1521/orcl]
SQL> insert into scott.emp (empno, ename) values (8000, 'CZARSKI');

1 Zeile wurde erstellt.

INSERTER@[sccloud033:1521/orcl]
SQL> insert into scott.emp (empno, ename, sal) values (8000, 'CZARSKI', 900);
insert into scott.emp (empno, ename, sal) values (8000, 'CZARSKI', 900)
                    *
FEHLER in Zeile 1:
ORA-01031: Nicht ausreichende Berechtigungen
```

Mehr als Rechte und Rollen

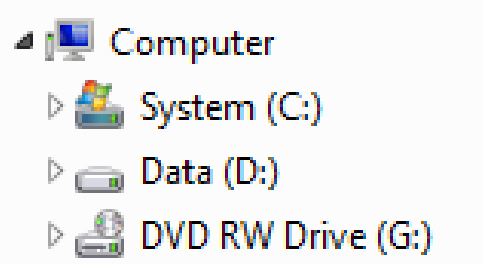
- Zeilenbasierte Zugriffsrechte
  - Virtual Private Database / Row Level Security
  - Enterprise Edition
- Arbeitsweise
  - Es wird ein komplettes SELECT-Privileg vergeben
  - Eine *Policy-Function* erweitert die SQL-Abfrage zur Laufzeit um zusätzliche Filter
  - Verschiedene User sehen verschiedene Zeilen
  - Auch spaltenbasiert möglich

Mehr als Rechte und Rollen
Ressourcenschutz für das Netzwerk

- Netzwerk-Security mit PL/SQL ACLs ab Oracle11g
- Zum Netzwerkzugriff braucht es ...
  - EXECUTE Privileg für das Package UTL_HTTP, UTL_TCP, DBMS_LDAP, UTL_SMTP, ...
  - Netzwerk-ACL zum Zugriff auf den jeweiligen Host

```
SQL> select * from USER_NETWORK_ACL_PRIVILEGES;

HOST                 LOWER_PORT UPPER_PORT PRIVILE STATUS
-------------------- ---------- ---------- ------- -------
*                                          connect GRANTED
*                                          resolve GRANTED

2 Zeilen ausgewählt.
```

Mehr als Rechte und Rollen
Ressourcenschutz für das Filesystem



- Directory Objekte
  - UTL_FILE, External Tables, Data Pump, ...
  - READ, WRITE und EXECUTE Privilegien
  - Zugriff auf alle Dateien im Verzeichnis, *nicht aber* auf Unterverzeichnisse

- Empfehlungen ...
  - Nur der DBA erzeugt Directories und vergibt Rechte
  - Niemals EXECUTE und WRITE-Privilegien, auf dem gleichen Directory, an den gleichen User vergeben

Weitere Informationen

- Dokumentation:
  - Oracle Security Guide
  - Oracle PL/SQL Developers Guide
- Deutschsprachige Community Seiten
  - DBA Community
    http://tinyurl.com/dbacommunity
  - APEX- und PL/SQL Community
    http://tinyurl.com/apexcommunity
  - Datenbankinfos auf dem Smartphone
    http://tinyurl.com/oraclebudb

Carsten.Czarski@oracle.com

http://tinyurl.com/apexcommunity

http://sql-plsql-de.blogspot.com
http://oracle-text-de.blogspot.com
http://oracle-spatial.blogspot.com

http://plsqlexecoscomm.sourceforge.net
http://plsqlmailclient.sourceforge.net

Twitter: @cczarski @oraclebudb