

Oracle Database Privilege Analysis

On the way to the "Least Privilege Principle" ...

Markus Flechtner

 @markusdba

 www.markusdba.net|.de

BASEL | BERN | BRUGG | BUCHAREST | DÜSSELDORF | FRANKFURT A.M. | FREIBURG I.BR. | GENEVA
HAMBURG | COPENHAGEN | LAUSANNE | MANNHEIM | MUNICH | STUTTGART | VIENNA | ZÜRICH

trivadis

Markus Flechtner

- Principal Consultant / Trivadis Germany GmbH
- Studied Mathematics a long time ago
- Focus
 - Oracle High Availability
 - Database Upgrade + Migration
- Teacher:
RAC, New Features, Multitenant, PostgreSQL



@markusdba



www.markusdba.net|.de



ORACLE
ACE

DOAG



BASEL | BERN | BRUGG | BUKAREST | DÜSSELDORF | FRANKFURT A.M. | FREIBURG I.B.R. | GENÈVE
HAMBURG | KOPENHAGEN | LAUSANNE | MANNHEIM | MÜNCHEN | STUTTGART | WIEN | ZÜRICH

trivadis



500+ technical experts helping peers globally

The **Oracle ACE Program** recognizes and rewards community members for their technical contributions in the Oracle community



3 membership tiers



For more details on Oracle ACE Program:
bit.ly/OracleACEProgram



Nominate
yourself or someone you know:
acenomination.oracle.com

Connect:  oracle-ace_ww@oracle.com

 Facebook.com/oracleaces

 [@oracleace](https://twitter.com/oracleace)



trivadis


**FOUNDED IN
1994**

250 SLA's
(SERVICE LEVEL AGREEMENTS)

 **700
EMPLOYEES**

 **16 TRIVADIS
WORKSPACES**
SWITZERLAND, GERMANY,
AUSTRIA, DENMARK,
ROMANIA

4000 
TRAINING PARTICIPANTS PER YEAR

**5 MILLION
CHF** 
BUDGET FOR SCIENCE
AND DEVELOPMENT PER YEAR

**118 MILLION
CHF
TURNOVER** 

800 
CUSTOMERS

**1900 PER
YEAR
PROJECTS**



trivadis

Agenda



- Introduction
- Package DBMS_PRIVILEGE_CAPTURE & Data Dictionary Objects
- Workflow
- Evaluation of the results and adopting the privileges
- Summary & Further Information

Introduction

History

- Security simply wasn't a focus for many legacy applications
- Many applications run with DBA-like privileges
- No privilege specification or analysis was performed at design time
- Focus was on getting the application running versus least privilege

```
SQL> grant DBA to PUBLIC with admin option;  
Grant succeeded.
```


Principle of the Least Privilege



"Every program and every privileged user of the system should operate using the least amount of privilege necessary to complete the job."

Jerome Salzer, Communications of the ACM, 1974

Oracle 12c introduced Privilege Analysis

- Captures the privileges which are used by an application resp. a database user
- Reports the used privileges (and the way ("path") the privileges have been granted)
- Reports the privileges which have been granted but have not be used
- Helps you to achieve the "Least Privilege Principle" for your own database applications
- However, there was this small note in the "Oracle Database Licensing Information":

Feature / Option / Pack	SE2	EE	EE-ES	DBCS SE	DBCS EE	DBCS EE-HP	DBCS EE-EP	ExaCS	Notes
Privilege Analysis	N	Y	Y	N	N	Y	Y	Y	EE and EE-ES: Requires the Oracle Database Vault option

November 2018: Licensing changed



Feature / Option / Pack	SE2	EE	EE-ES	DBCS SE	DBCS EE	DBCS EE-HP	DBCS EE-EP	ExaCS	Notes
Privilege Analysis	N	Y	Y	N	Y	Y	Y	Y	

- Privilege Analysis is now available for Oracle Database Enterprise Edition (for all versions since Oracle Database 12c Release 1), **Database Vault is not required anymore**

Of course, it's not that easy ..

- Logging database usage is a kind of auditing
 - Especially when using personalized accounts
 - Oracle Privilege Analysis captures which privileges were used but not the exact time when they were used (you can only determine the time range = time when the analysis ran)
 - You may be required to ask the workers council for an approval
 - But security is a strong argument
- Expect resistance
 - From 3rd party software vendors
 - From your own developers

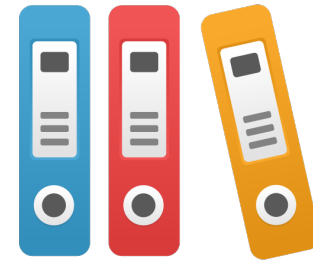
trivadis



.. but it will help you as the DBA

- Required privileges will be documented
- **High privileges which are not used (required) are documented**
- Raise security concerns, tell your manager
 - Then (s)he's in charge

trivadis



Package DBMS_PRIVILEGE_CAPTURE & Data Dictionary Objects

Package DBMS_PRIVILEGE_CAPTURE



Procedure	Purpose
CREATE_CAPTURE	Defines a capture policy
ENABLE_CAPTURE	Starts a privilege capture run
DISABLE_CAPTURE	Ends a privilege capture run
GENERATE_RESULT	Fills the result views with the results of a capture run
DROP_CAPTURE	Drops a capture policy and the associated results
DELETE_RUN	Deletes the results of a capture run (but not the policy)
CAPTURE_DEPENDENCY_PRIVS	Captures the privileges that are used by definer's rights and invoker's rights PL/SQL program units for compilation (has to be enabled manually after a capture was started)

Data Dictionary Views & Internal Tables

Name	Purpose / Content
DBA_PRIV_CAPTURES	defined capture policies and runs (via "DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTURE")
PRIV_CAPTURE\$	(basis of DBA_PRIV_CAPTURES)
CAPTURED_PRIV\$	Captured privileges (*)
CAPTURE_RUN_LOG\$	Information on the capture runs (*) Contains information on start time and end time (which is not visible in the DBA_%-views)

- (*) Both result tables are located in the SYSAUX tablespace

Result Views (1)

Views for used privileges	Views for unused privileges
Overview (all privileges resp. grants)	
DBA_USED_PRIVS	DBA_UNUSED_PRIVS
	DBA_UNUSED_GRANTS
Privileges granted to Public	
DBA_USED_PUBPRIVS	DBA_UNUSED_PUBPRIVS
System Privileges	
DBA_USED_SYSPRIVS	DBA_UNUSED_SYSPRIVS
DBA_USED_SYSPRIVS_PATH	DBA_UNUSED_SYSPRIVS_PATH

Result Views (2)

Views for used privileges	Views for unused privileges
Object Privileges	
DBA_USED_OBJPRIVS	DBA_UNUSED_OBJPRIVS
DBA_USED_OBJPRIVS_PATH	DBA_UNUSED_OBJPRIVS_PATH
User Privileges	
DBA_USED_USERPRIVS	DBA_UNUSED_USERPRIVS
DBA_USED_USERPRIVS_PATH	DBA_UNUSED_USERPRIVS_PATH

- CDB_%-Views are available, too.

Workflow

Define Capture Policy (1) – What to capture?



- You must know, how to identify the application in the database, e.g.
 - Specific user
 - Role(s) granted to the user which is used by the application
 - Session context
- Based on that you can define the capture policy
- Possible capture types
 - All database activities
 - Validate role privileges by capturing all privileges which are included in a role or a set of roles
 - Database sessions which fulfill certain context conditions (function SYS_CONTEXT)

Define Capture Policy (2) – What to capture?

- Constants in DBMS_PRIVILEGE_CAPTURE (for capture type):

G_DATABASE	capture all database activities (resp. used privileges) except for SYS activities
G_ROLE	captures privilege use of one ore more roles
G_CONTEXT	captures all privilege use in a specified context
G_ROLE_AND_CONTEXT	combination of G_ROLE and G_CONTEXT

Define Capture Policy (3) – CREATE_CAPTURE

Procedure **DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTURE**

Argument Name	Type	In/Out	Default?
-----	-----	-----	-----
NAME	VARCHAR2	IN	
DESCRIPTION	VARCHAR2	IN	DEFAULT
TYPE	NUMBER	IN	DEFAULT
ROLES	ROLE_NAME_LIST	IN	DEFAULT
CONDITION	VARCHAR2	IN	DEFAULT

- "CONDITION" has to be used to define the context for the capture types "G_CONTEXT" and "G_ROLE_AND_CONTEXT"

Define Capture Policy (4) - Examples

REM policy to capture all database activities

```
execute DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTURE(  
  name => 'POLICY_ALL_DB_ACTIVITIES',  
  description => 'captures all database privileges used by all users',  
  type => DBMS_PRIVILEGE_CAPTURE.G_DATABASE  
);
```

REM which PUBLIC privileges are used by an application/user

```
execute DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTURE(  
  name => 'POLICY_CAPTURE_PUBLIC',  
  description => 'captures all required privileges granted to public',  
  type => DBMS_PRIVILEGE_CAPTURE.G_ROLE,  
  roles => 'PUBLIC'  
);
```

Define Capture Policy (5) - Examples

REM which privileges are used by a specific user

```
execute DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTURE(  
  name => 'POLICY_CAPTURE_SCOTT',  
  description => 'captures the privileges required by SCOTT',  
  type => DBMS_PRIVILEGE_CAPTURE.G_CONTEXT,  
  condition=> q'[sys_context('USERENV','SESSION_USER') = 'SCOTT']'  
);
```

REM which DBA privileges are used by a specific user

```
execute DBMS_PRIVILEGE_CAPTURE.CREATE_CAPTURE(  
  name => 'POLICY_CAPTURE_SCOTT_DBA',  
  description => 'captures all required privileges granted to public',  
  type => DBMS_PRIVILEGE_CAPTURE.G_ROLE_AND_CONTEXT,  
  roles => 'DBA',  
  condition=> q'[sys_context('USERENV','SESSION_USER') = 'SCOTT']'  
);
```


Define Capture Policy (6) - SYS_CONTEXT

- SYS_CONTEXT is the only function which can be used to specify the conditions for "DBMS_PRIVILEGE_CAPTURE.G_CONTEXT"
- No user defined functions (but you can use a user defined context)

Examples:

SESSION_USER	User who logged in
HOST	Client machine
OS_USER	Client OS User
MODULE	via DBMS_APPLICATION_INFO
ACTION	via DBMS_APPLICATION_INFO
User defined context	via DBMS_SESSSION.SET_CONTEXT

Start Privilege Capture

- Start privilege capture

PROCEDURE DBMS_PRIVILEGE_CAPTURE.ENABLE_CAPTURE

Argument Name	Type	In/Out	Default?
-----	-----	-----	-----
NAME	VARCHAR2	IN	
RUN_NAME	VARCHAR2	IN	DEFAULT

- For one profile multiple test runs can be stored
- Enable capture of dependency privileges if required
- Example for starting a privilege capture

```
Execute DBMS_PRIVILEGE_CAPTURE.ENABLE_CAPTURE (  
  name => 'POLICY_CAPTURE_SCOTT',  
  run_name => 'TEST_RUN_20191110');
```

Run your Application



- **That's the critical part**
- You have to run all modules, screen, batch jobs etc. which are ever used by your application
- Hopefully you have got a **complete (!) set of automated (!) tests**
- Missing a function which runs e.g. once a year and which requires a special privilege will cause this function to fail (some time later) if you adopt the privileges according to the results of the privilege capture!

Stop Privilege Capture

- After the tests are complete the capture can be stopped

```
PROCEDURE DBMS_PRIVILEGE_CAPTURE.DISABLE_CAPTURE
Argument Name          Type          In/Out Default?
-----
NAME                    VARCHAR2      IN
```

- Example:

```
Execute DBMS_PRIVILEGE_CAPTURE.DISABLE_CAPTURE (
  name => 'POLICY_CAPTURE_SCOTT' );
```

Fill Result Views (1)

- The results which are stored in internal tables after the run has been stopped have to be transferred into the DBA_USED_%- and DBA_UNUSED_%-views

```
PROCEDURE DBMS_PRIVILEGE_CAPTURE.GENERATE_RESULT
Argument Name      Type              In/Out  Default?
-----
NAME               VARCHAR2         IN
RUN_NAME          VARCHAR2         IN      DEFAULT
DEPENDENCY         BOOLEAN          IN      DEFAULT
```

- Setting DEPENDENCY=TRUE is required when capturing dependent privileges (CAPTURE_DEPENDENCY_PRIVS)

Fill Result Views (2)

- Example:

```
Execute DBMS_PRIVILEGE_CAPTURE.GENERATE_RESULT (  
  name => 'POLICY_CAPTURE_SCOTT',  
  run_name => 'TEST_RUN_20191110');
```

- The run_name must be the same as the one you specified when you enabled the capture
- If you do not specify the run_name, the capture will be stopped but the column "RUN_NAME" in the result table will be empty.

Miscellaneous (1)

- The role `CAPTURE_ADMIN` is required to run procedures of the package `DBMS_PRIVILEGE_CAPTURE`
- Only one privilege capture policy can be active at a time
- Enabled capture policies remain active even after a restart of the database instance
 - But the privileges which were captured before the restart are lost ☹
- Results are stored until the run is deleted (`DBMS_PRIVILEGE_CAPTURE.DELETE_RUN`) or the policy is dropped (`DBMS_PRIVILEGE_CAPTURE.DROP_POLICY`)
 - Create your own result tables via CTAS to avoid the loss of data
- In a Container Database you can run privilege analysis on container level only (`CDB$ROOT` and individual PDBs), not globally for all containers
- The performance impact of privilege capture can be neglected (at least according to my experience)

Miscellaneous (2)

- If you consider the required space in tablespace SYSAUX as an issue, the following workflow may be an option
- Run DBMS_PRIVILEGE_CAPTURE on a daily basis (e.g. via database job)
 - ..
 - disable_capture ('policy','next_run')
 - Generate result ('policy','current_run')
 - Insert into own_table select * from dba_used/unused where run_name='current_run'
 - Delete_run ('policy','current_run')
 - enable capture ('policy','current_run')
 - ..
- Of course, there's the risk that special privileges which are used between disabling / enabling the policy are not captured
- The same procedure may help preventing loss of capture data due to a restart of an instance.

Miscellaneous (3)

- When using objects from another schema for own objects, e.g. views or PL/SQL code, granting privileges via a role is not sufficient: Direct grants are required
- Assuming a user has been granted a privilege both via role and directly and a direct grant is required, this will be reflected in DBA_USED_PRIVS (USERNAME=USED_ROLE)

```
SELECT run_name, object_owner, object_name, username, used_role
FROM dba_used_privs WHERE object_owner = 'HR';
```

⚡ RUN_NAME	⚡ OBJECT_OWNER	⚡ OBJECT_NAME	⚡ USERNAME	⚡ USED_ROLE
C_DIRECTVIEW	HR	EMPLOYEES	PAUSER	PAUSER
A_ROLEONLY	HR	EMPLOYEES	PAUSER	PAUSER_ROLE

Direct grant was required
for creating a view

SELECT only, role grant
was sufficient

- When granting privileges (after the analysis), joining the results with DBA_DEPENDENCIES may be beneficial, too.

Evaluating the results & Adopting the privileges

Example Result Queries (1)

- Which system privileges were used and how were they granted? ("grant path")

```
SQL> select USED_ROLE, SYS_PRIV, PATH
       2 from DBA_USED_SYSPRIVS_PATH where CAPTURE='POLICY_CAPTURE_SCOTT'
       3 and RUN_NAME='TEST_RUN_20191110';
```

USED_ROLE	SYS_PRIV	PATH
TOP_SECRET	SELECT ANY TABLE	GRANT_PATH('SCOTT')
TOP_SECRET	SELECT ANY TABLE	GRANT_PATH('SCOTT', 'SECRET', 'TOP_SECRET')
TOP_SECRET	ANALYZE ANY	GRANT_PATH('SCOTT', 'SECRET', 'TOP_SECRET')
CONNECT	CREATE SESSION	GRANT_PATH('SCOTT', 'CONNECT')

The privilege "SELECT ANY TABLE" was granted in two ways

Role TOP_SECRET was granted to the role SECRET and the role SECRET was granted to SCOTT

Example Result Queries (2)

- Which object privileges were used?

```
SQL> select USERNAME,USED_ROLE,OBJ_PRIV,  
2  OBJECT_OWNER O_OWNER,OBJECT_TYPE O_TYPE,OBJECT_NAME O_NAME  
3  from DBA_USED_OBJPRIVS  
4  where CAPTURE='POLICY_CAPTURE_SCOTT'  
5  and RUN_NAME= 'TEST_RUN_20191110';
```

USERNAME	USED_ROLE	OBJ_PRIV	O_OWNER	O_TYPE	O_NAME
-----	-----	-----	-----	-----	-----
SCOTT	PUBLIC	EXECUTE	SYS	PACKAGE	DBMS_APPLICATION_INFO
SCOTT	PUBLIC	SELECT	SYS	TABLE	DUAL
SCOTT	SCOTT	SELECT	HR	TABLE	DEPARTMENTS
SCOTT	SCOTT	SELECT	HR	TABLE	EMPLOYEES
SCOTT	SCOTT	UPDATE	HR	TABLE	EMPLOYEES

Example Result Queries (3)

- All privileges which were used during the privilege analysis capture

```
SQL> select OBJ_PRIV,SYS_PRIV,OBJECT_OWNER O_OWNER,
  2  OBJECT_NAME O_NAME,OBJECT_TYPE O_TYPE from DBA_USED PRIVS
  3  where CAPTURE='POLICY_CAPTURE_SCOTT' and RUN_NAME= 'TEST_RUN_20191110';
```

OBJ_PRIV	SYS_PRIV	O_OWNER	O_TYPE	O_NAME
UPDATE		HR	TABLE	EMPLOYEES
	SELECT ANY TABLE	HR	TABLE	EMPLOYEES
	SELECT ANY TABLE	HR	TABLE	EMPLOYEES
SELECT		HR	TABLE	DEPARTMENTS
	ANALYZE ANY	HR	TABLE	EMPLOYEES
SELECT		SYS	TABLE	DUAL
SELECT		HR	TABLE	EMPLOYEES
	CREATE SESSION			
EXECUTE		SYS	PACKAGE	DBMS_APPLICATION_INFO

Example Result Queries (4)

- All privileges which were granted to the to the user SCOTT but not used during the privilege analysis capture

```
SQL> select OBJ_PRIV,SYS_PRIV,OBJECT_OWNER O_OWNER,
  2  OBJECT_NAME O_NAME,OBJECT_TYPE O_TYPE from DBA UNUSED PRIVS
  3  where CAPTURE='POLICY_CAPTURE_SCOTT' and RUN_NAME= 'TEST_RUN_20191110';
```

OBJ_PRIV	SYS_PRIV	O_OWNER	O_TYPE	O_NAME

SELECT ANY DICTIONARY				
EXECUTE		SYS	DIRECTORY	DATA_PUMP_DIR
READ		SYS	DIRECTORY	DATA_PUMP_DIR
WRITE		SYS	DIRECTORY	DATA_PUMP_DIR
SELECT		SYS	VIEW	V_\$SQL_PLAN_STATISTICS_ALL
SELECT		SYS	VIEW	V_\$SESSION
SELECT		SYS	VIEW	V_\$SQL_PLAN
SELECT		SYS	VIEW	V_\$SQL
EXECUTE		SYS	PACKAGE	DBMS_FLASHBACK_ARCHIVE
EXECUTE		SYS	PACKAGE	DBMS_FLASHBACK
EXECUTE		SYS	PACKAGE	DBMS_MONITOR
SELECT		HR	TABLE	LOCATIONS
UPDATE		HR	TABLE	LOCATIONS
[...]				

Example Result Queries (5)



- The views contain much more information



```
SQL> desc DBA_USED_PRIVS
```

Name	Null?	Type
CAPTURE	NOT NULL	VARCHAR2 (128)
SEQUENCE	NOT NULL	NUMBER
OS_USER		VARCHAR2 (128)
USERHOST		VARCHAR2 (128)
MODULE		VARCHAR2 (64)
USERNAME	NOT NULL	VARCHAR2 (128)
USED_ROLE		VARCHAR2 (128)
SYS_PRIV		VARCHAR2 (40)
OBJ_PRIV		VARCHAR2 (40)
USER_PRIV		VARCHAR2 (25)
OBJECT_OWNER		VARCHAR2 (128)
OBJECT_NAME		VARCHAR2 (128)
OBJECT_TYPE		VARCHAR2 (23)
COLUMN_NAME		VARCHAR2 (128)
OPTION\$		NUMBER
PATH		GRANT_PATH
RUN_NAME		VARCHAR2 (128)


Privilege Analysis & OEM Cloud Control (1)

- OEM Cloud Control 13c can be used to manage Privilege Analysis
 - Targets → Database → Security → Privilege Analysis

 **OEMREP.markusdba.local** 

Logged in as **system**  |  **kaki.markusdba.local**

Oracle Database ▼ Performance ▼ Availability ▼ Security ▼ Schema ▼ Administration ▼







Page Refreshed Nov 17, 2019 10:40:26 PM CET 

Privilege Analysis

Privilege Analysis enables you to find information about privilege usage for a database according to a specified condition, such as privileges to run an application module or privileges used in a given user session. It analyzes both system privileges and object privileges. To monitor the privileges used for a user's action, you must create and enable a Privilege Analysis policy. Afterward, you can generate a report that describes the used and unused privileges and then from there, revoke (and regrant) privileges as necessary. However, you cannot use privilege analysis to analyze the use of SYS user privileges. Privilege analysis is licensed as part of Oracle Database Vault, but you do not need to enable Database Vault to use it.

Policies

Actions ▼ View ▼

 **Create Capture**  **Start Capture**  **Stop Capture**  **Generate Report**  **View Reports**  **Delete Capture**

Policy	Active	Capture Scope					Unused Privileges		Scheduled Jobs	
		Type	First Start Time	Last End Time	Total Capture Duration	Users	System	Object		
ORA\$DEPENDENCY		Database								
scott_privilege_analysis		Context	Nov 17, 2019 10:40...	Nov 17, 2019 10:40...		1	9	4		

Privilege Analysis & OEM Cloud Control (2)

Oracle Database ▼ Performance ▼ Availability ▼ Security ▼ Schema ▼ Administration ▼

Page Refreshed Nov 17, 2019 10:40:57 PM CET ↺

Privilege Analysis: Reports

Summary Unused Used

The usage report provides a hierarchical representation of each unused and used privilege, and the grant path. From here, you can revoke and regrant privileges and roles to and from users as necessary.

▲ Search

Policy * Grantee 🔍

✕ Revoke

↺ Regrant

Grantee	Type	Used	Revoked	System Privileges		Object Privileges		
				Unused	Used	Unused	Used	
▲ SCOTT	User			9	3	4	6	
▲ Object Privileges	Folder					2	2	
▲ HR	Schema					2	2	
▲ TABLE	Object Type					2	2	
▶ EMP_TEST2	Object Name						2	
▶ LOCATIONS	Object Name					2		
▶ CONNECT	Role	✓		1	1			
▶ RESOURCE	Role			8		2		
▶ SECRET	Role	✓						

Generate "GRANT" commands (1)

- The results of a privilege capture can be used to create a GRANT-script
- Example: GRANT all required privileges to a new role SCOTT_ROLE
- Part 1: System Privileges

```
SQL> SELECT 'grant '||sys_priv||' to SCOTT_ROLE;' PRIVS_TO_GRANT
2  FROM DBA_USED_PRIVS where SYS_PRIV not like '%ANY%'
3  and CAPTURE='POLICY_CAPTURE_SCOTT'
4  and RUN_NAME= 'TEST_RUN_20191110';
```

Generate "GRANT" commands (2)

- Part 2. Object Privileges

```
SQL> SELECT DISTINCT 'grant ' ||  
 2 CASE SYS_PRIV  
 3     WHEN 'SELECT ANY TABLE' THEN 'SELECT'  
 4     WHEN 'EXECUTE ANY PROCEDURE' THEN 'EXECUTE'  
 5     WHEN 'INSERT ANY TABLE' THEN 'INSERT'  
 6     WHEN 'UPDATE ANY TABLE' THEN 'UPDATE'  
 7     WHEN 'DELETE ANY TABLE' THEN 'DELETE'  
 8     WHEN 'ANALYZE ANY' THEN 'ANALYZE'  
 9     WHEN 'SELECT ANY SEQUENCE' THEN 'SELECT'  
10 ELSE  
11     OBJ_PRIV  
12 END  
13 || ' on ' || OBJECT_OWNER || '.' || OBJECT_NAME || ' to SCOTT_ROLE;' PRIVS_TO_GRANT  
14 FROM DBA_USED_PRIVS where object_name is not null;
```

- Query originally based on <https://apex.oracle.com/pls/apex/germancommunities/dbacomcommunity/tipp/7141/index.html>
Credits to Norman Sibbing from Oracle

Generate "GRANT" commands (3) - Result

```
grant CREATE SESSION to SCOTT_RESTRICTED_PRIVS_ROLE;  
  
grant EXECUTE on SYS.DBMS_APPLICATION_INFO to SCOTT_RESTRICTED_PRIVS_ROLE;  
grant SELECT on HR.EMPLOYEES to SCOTT_RESTRICTED_PRIVS_ROLE;  
grant SELECT on HR.EMP_TEST2 to SCOTT_RESTRICTED_PRIVS_ROLE;  
grant SELECT on HR.DEPARTMENTS to SCOTT_RESTRICTED_PRIVS_ROLE;  
grant ANALYZE on HR.EMPLOYEES to SCOTT_RESTRICTED_PRIVS_ROLE;  
grant UPDATE on HR.EMP_TEST2 to SCOTT_RESTRICTED_PRIVS_ROLE;  
grant SELECT on SYS.DUAL to SCOTT_RESTRICTED_PRIVS_ROLE;
```

Summary & Further Information

Summary



- "Privilege Analysis" is a great tool for achieving the "Principle of the Least Privilege"
- Privilege Analysis should be included in your tests
- It's critical that you run all functions, modules, batch jobs etc. of your application during the capture phase (Automation can help 😊)
- Lifting the license restrictions (Database Vault) was an important step made by Oracle to help the customers making their applications more secure

Further Information

- Wikipedia: "Principle of least privilege": https://en.wikipedia.org/wiki/Principle_of_least_privilege
- Documentation of the package DBMS_PRIVILEGE_CAPTURE:
https://docs.oracle.com/en/database/oracle/oracle-database/19/arpls/DBMS_PRIVILEGE_CAPTURE.html#GUID-6522AC3E-A457-4C7B-8996-B065957F73E4
- Database Security Guide, Chapter 5 "Performing Privilege Analysis to Find Privilege Use":
<https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/performing-privilege-analysis-find-privilege-use.html#GUID-44CB644B-7B59-4B3B-B375-9F9B96F60186>
- Deutschsprachiger Datenbank & Cloud Technologie Blog:
"Least Privileges mit Oracle Privilege Analysis"
<https://blogs.oracle.com/coretec/least-privileges-mit-oracle-privilege-analysis>
- MOS-Note "Privilege Analysis Feature of Database Vault (Doc ID 2588251.1)"
- <https://gavinsoorma.com/2015/02/oracle-12c-new-feature-privilege-analysis/>

Questions & Answers

Markus Flechtner

markus.flechtner@trivadis.com

Phone +49 211 5866 64725



@markusdba



www.markusdba.net|.de

BASEL | BERN | BRUGG | BUKAREST | DÜSSELDORF | FRANKFURT A.M. | FREIBURG I.B.R. | GENÈVE
HAMBURG | KOPENHAGEN | LAUSANNE | MANNHEIM | MÜNCHEN | STUTTGART | WIEN | ZÜRICH

trivadis



Making a **WORLD** possible
in which **intelligent IT**
facilitates **LIFE and WORK** as a
matter of course.