Example 3: Implementing column-level access policies

Overview

In row-level access policies, the policy is applied regardless of the selected columns. However, when implementing restrictions at the column level, the policy is not enforced until the columns protected by the policy are included in the DML statement. As we will see, this option can also be used to mask column data when desired. When column masks are also enforced by the policy, the records that don't conform to the defined criteria have their column values hidden by the policy and displayed as nulls instead.

Getting ready

In this recipe we will create two users; DOCONNEL and JWHALEN. We will declare a policy named EMPLOYEES_SALCOMM_PLC that will protect the salary and commision_pct columns from the EMPLOYEES_TEST_VPD table. Then we will redefine the VPD policy to apply column masking.

Workflow

1. As system user create users DOCONNEL and JWHALEN:

```
create user DOCONNEL identified by DOCONNEL;
create user JWHALEN identified by JWHALEN;
grant create session to DOCONNL,JWHALEN;
```

2. As user HR grant select on employee table as follows:

```
grant select on hr.employees_test_vpd to DOCONNEL,JWHALEN;
```

3. Connect as security_adm user and create salcomm_plc_func policy function:

```
CREATE OR REPLACE
FUNCTION salcomm_plc_func
  (
    schema_v IN VARCHAR2,
    tbl_v VARCHAR2)
  RETURN VARCHAR2
IS
  ret_val VARCHAR2(200);
BEGIN
  ret_val := 'email =  SYS_CONTEXT(''USERENV'', ''SESSION_USER'')';
  RETURN ret_val;
END;
```

4. Create the column-level policy EMPLOYEES_SALCOMM_PLC as follows:

```
BEGIN
    DBMS_RLS.add_policy
 (
  object_schema => 'HR',
  object_name => 'EMPLOYEES_TEST_VPD',
  policy_name => 'employees_salcomm_plc',
policy_function => 'salcomm_plc_func',
statement_types => 'SELECT',
sec_relevant_cols => 'SALARY,COMMISSION_PCT'
);
END;
```

5. Connect as user DOCONNEL and issue a select statement without including the protected columns salary and commission_pct as follows:

```
select first_name,last_name from hr.employees_test_vpd;
```

6. Now issue a select statement that includes the salary column as follows:

```
select first_name,last_name,salary from hr.employees;
```

7. Next issue a select statement that includes the commission_pct column as follows:

```
select first_name,last_name,commission_pct from hr.employees;
```

8. And finally include salary and also commission_pct column as follows:

```
select first_name,last_name,salary,commission_pct from hr.employees;
```

9. Connect as user JHWALEN and repeat some of the statements from that performed for DOCONNEL user . Connect as user security_adm and disable the policy employees_salcomm_plc:

```
conn security_adm
Enter password:
 BEGIN
  2
dbms_rls.enable_policy(policy_name=>'employees_salcomm_plc',object_name=>'employe
es_test_vpd', object_schema=>'HR',enable=>FALSE);
  3    END;
  4  /
```

10. Create a new policy named employee_salcomm_plc_mask using the data masking option:

```
begin
  2      DBMS_RLS.add_policy (object_schema => 'HR', object_name =>
'EMPLOYEES_TEST_VPD', policy_name => 'employees_salcomm_plc_mask',
policy_function => 'salco
mm_plc_func', statement_types => 'SELECT', sec_relevant_cols =>
'SALARY,COMMISSION_PCT', sec_relevant_cols_opt => DBMS_RLS.all_rows );
  3  end;
  4  /
```

11. Connect as user DOCONNEL and issue the following statement:

```
select first_name,last_name,salary,commission_pct fromhr.employees;
```

The salary and commission_pct has values just for the user DOCONNEL; for other users' salary and commission_pct are displayed as null.

How it works

The policy will not trigger unless the columns declared in sec_relevant_cols are not used in statements. Column masking works only with SELECT statements. Additional information about secured columns can be found in the DBA_SEC_RELEVANT_COLS dictionary view.