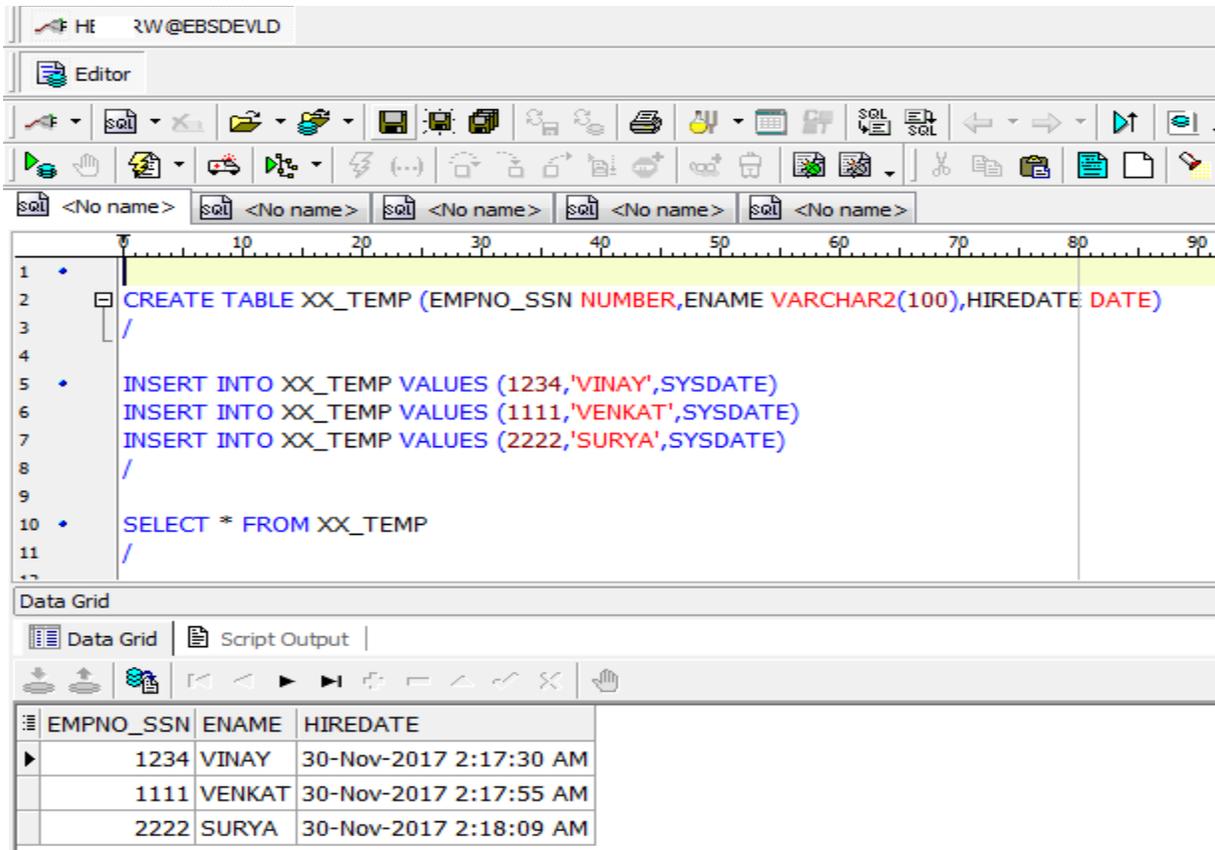


## Masking the Data for a Column for a Specific Schema using the RLS Policy

Worked on a specific requirement were in had to Mask/Hide the Data of a specific column while the data was been queried from the Table from a specific schema.

The same is been explained by creating a Table in One Schema which some specific columns here we consider two schema one is our Custom Schema and the other is APPS Schema as below:

**Note:** Table, Function, DBMS\_RLS is executed in Custom Schema



```
1 CREATE TABLE XX_TEMP (EMPNO_SSN NUMBER, ENAME VARCHAR2(100), HIREDATE DATE)
2 /
3
4
5 INSERT INTO XX_TEMP VALUES (1234, 'VINAY', SYSDATE)
6 INSERT INTO XX_TEMP VALUES (1111, 'VENKAT', SYSDATE)
7 INSERT INTO XX_TEMP VALUES (2222, 'SURYA', SYSDATE)
8 /
9
10 SELECT * FROM XX_TEMP
11 /
```

EMPNO_SSN	ENAME	HIREDATE
1234	VINAY	30-Nov-2017 2:17:30 AM
1111	VENKAT	30-Nov-2017 2:17:55 AM
2222	SURYA	30-Nov-2017 2:18:09 AM

Now we try to Mask the Data of EMPNO\_SSN Column of XX\_TEMP Table while it is been queried from APPS Schema.

```
CREATE OR REPLACE FUNCTION XX_TEMP_FUNC (p_owner IN VARCHAR2, p_name IN VARCHAR2)
RETURN VARCHAR2
AS
BEGIN
IF SYS_CONTEXT ('userenv', 'session_user') = 'HR' -- Custom Schema Name
THEN
RETURN NULL;
ELSE
RETURN '1=0';
END IF;
END;
```

Setting the Policy using the below standard defined Package

```
BEGIN
```

```
  DBMS_RLS.ADD_POLICY(object_schema=>'H████RW',  
    object_name=>'XX_TEMP', --Name of table, view, or synonym to which the policy is added.  
    policy_name=>'SEC_SSN',  
    function_schema=>'H████RW',  
    policy_function=>'XX_TEMP_FUNC', --Name of a function which generates a predicate for the policy.  
    sec_relevant_cols=>'EMPNO_SSN',  
    sec_relevant_cols_opt=> dbms_rls.ALL_ROWS);
```

```
END;
```

```
/
```

**Note:** Any change to the above policy we need to first drop the Policy and then re-create it. The details of the Policy is stored in ALL\_POLICIES Table.

For the table that we have created we need to give Grants and Synonyms in APPS Schema

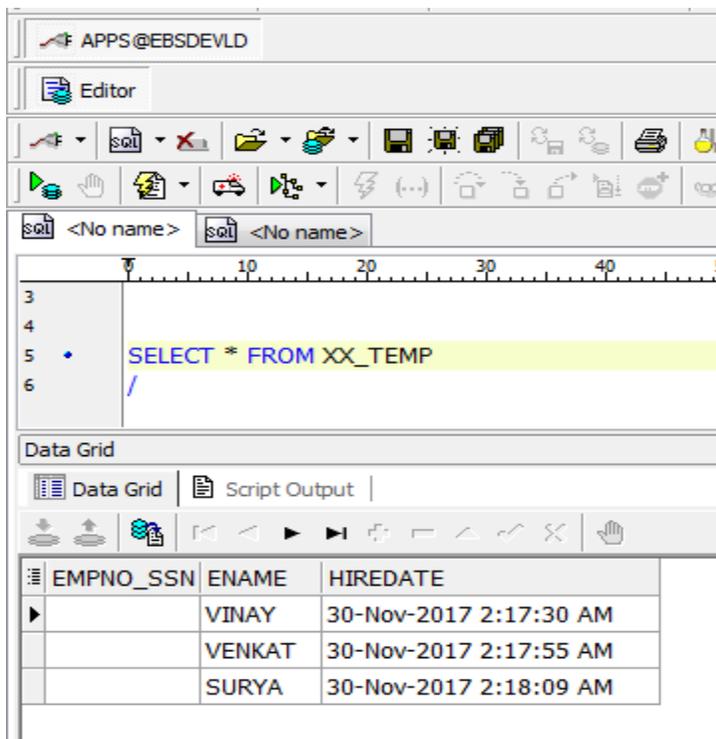
```
GRANT ALL ON XX_TEMP TO APPS
```

```
/
```

```
CREATE SYNONYM APPS.XX_TEMP FOR H████RW.XX_TEMP
```

```
/
```

Once all the above steps are done when we query the table in APPS here what we see



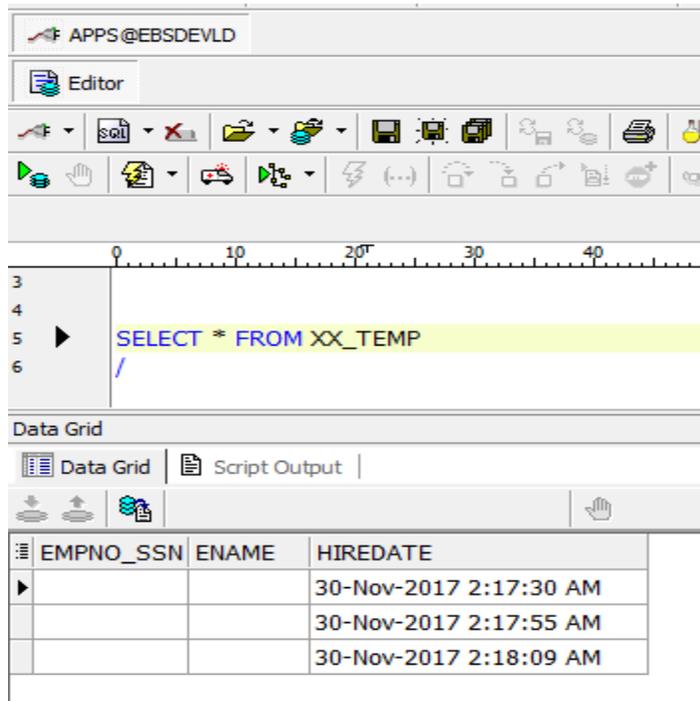
The screenshot shows the Oracle SQL Developer interface. The top toolbar includes icons for file operations, SQL execution, and debugging. The main editor window displays the following SQL query:

```
3  
4  
5 • SELECT * FROM XX_TEMP  
6 /
```

Below the editor is the 'Data Grid' section, which shows the results of the query. The results are displayed in a table with three columns: EMPNO\_SSN, ENAME, and HIREDATE. The data is as follows:

EMPNO_SSN	ENAME	HIREDATE
	VINAY	30-Nov-2017 2:17:30 AM
	VENKAT	30-Nov-2017 2:17:55 AM
	SURYA	30-Nov-2017 2:18:09 AM

Now masking can also be applied to multiple columns as well.



The screenshot displays the Oracle SQL Developer interface. At the top, the user is identified as 'APPS@EBSDEVLD'. The main window is titled 'Editor' and contains a SQL query: `SELECT * FROM XX_TEMP`. Below the editor, the 'Data Grid' is visible, showing the results of the query. The grid has three columns: 'EMPNO\_SSN', 'ENAME', and 'HIREDATE'. The first column is masked with a black box. The 'HIREDATE' column shows three rows of data, all with the same timestamp: '30-Nov-2017 2:17:30 AM', '30-Nov-2017 2:17:55 AM', and '30-Nov-2017 2:18:09 AM'.

EMPNO_SSN	ENAME	HIREDATE
		30-Nov-2017 2:17:30 AM
		30-Nov-2017 2:17:55 AM
		30-Nov-2017 2:18:09 AM

This ends the Masking/Hiding of Column Data for a specific schema.

Thanks!!!!!!