



# FINE-GRAINED ACCESS CONTROL

CHỦ ĐỀ 6.3 – GIẢNG VIÊN: TUẤN NGUYỄN HOÀI ĐỨC

## MỤC LỤC

I.	GIỚI THIỆU NHÓM + PHÂN CÔNG CÔNG VIỆC .....	3
a.	THÀNH VIÊN .....	3
b.	PHÂN CÔNG .....	3
II.	VIRTUAL PRIVATE DATABASE (VPD).....	4
a.	KHÁI NIỆM VPD .....	4
b.	COLUMN-LEVEL VPD VÀ COLUMN MASKING.....	6
1.	COLUMN LEVEL VPD .....	6
2.	COLUMN MASKING .....	7
c.	VPD SECURITY POLICY .....	8
1.	DBMS_RLS PROCEDURES .....	8
2.	ÁP DỤNG CÁC POLICIES TRÊN TABLE, VIEW, SYNONYM .....	8
3.	TỔ CHỨC POLICIES THEO GROUP .....	14
4.	CÁC POLICIES MẶC ĐỊNH.....	18
d.	CÀI ĐẶT VPD ĐỂ CÓ FINE-GAINED ACCESS CONTROL .....	20
e.	GIỚI HẠN PHẠM VI SỬ DỤNG ĐƯỢC VPD (NHỮNG SCHEMA NÀO) .....	25
III.	KHÁI NIỆM APPLICATION CONTEXT .....	27
a.	CÁC ĐẶC TRƯNG CỦA APPLICATION CONTEXT VÀ CÁCH THIẾT LẬP.....	27
1.	KHÁI NIỆM .....	27
2.	CÁC ĐẶC TRƯNG .....	27
3.	LỢI ÍCH .....	28
b.	SESSION-BASED/LOCAL/GLOBAL APPLICATION CONTEXT.....	28
1.	CLIENT SESSION-BASED APPLICATION CONTEXT (NGŨ CẢNH ỨNG DỤNG DỰA TRÊN LẦN ĐĂNG NHẬP CỦA KHÁCH HÀNG).....	28
2.	DATABASE SESSION-BASED APPLICATION CONTEXTS (NGŨ CẢNH ỨNG DỤNG DỰA TRÊN LẦN LÀM VIỆC).....	28
3.	GLOBAL APPLICATION CONTEXT (NGŨ CẢNH ỨNG DỤNG CHUNG).....	29
c.	CÁCH DÙNG APPLICATION CONTEXT CHO FINE-GAINED ACCESS CONTROL .....	30
d.	KIỂM TRA TÁC DỤNG POLICY: VIEW VS VPD_POLICY .....	32
IV.	DEMO VPD, APPLICATION CONTEXT VÀ VPD.....	33
a.	VPD.....	33
1.	TÌNH HUỐNG .....	33
2.	CÁC CHÍNH SÁCH CẦN THIẾT LẬP CHO VPD .....	35
3.	CHÍNH SÁCH VPD ĐẦU TIÊN.....	36



4. CHÍNH SÁCH VPD THỨ HAI .....	39
b. APPLICATION CONTEXT .....	43
1. VẤN ĐỀ.....	43
2. TẠO APPLICATION CONTEXT CẦN TẠO CHO DEMO.....	43
3. ÁP DỤNG APPLICATION CONTEXT VÀO VPD .....	47
4. CHI SẺ CHO TẤT CẢ NGƯỜI DÙNG APPLICATION CONTEXT GLOBAL .....	50
V. NGUỒN THAM KHẢO .....	52

## I. GIỚI THIỆU NHÓM + PHÂN CÔNG CÔNG VIỆC

### A. THÀNH VIÊN

STT	MSSV	TÊN
1	1512454	PHAN THANH SANG
2	1512168	NGUYỄN THẾ HIỀN
3	1412653	ĐẶNG THỊ THÙY VY
4	1512417	PHÙNG KHẮC PHƯƠNG
5	1412006	HỨA TUẤN ANH

### B. PHÂN CÔNG

STT	CÔNG VIỆC	MSSV
1	Khái niệm VPD	1512417
2	Column-level VPD, column masking	1512454, 1512417
3	VPD Security Policy: <ul style="list-style-type: none"> <li>Áp dụng các Policies trên table, view, synonym</li> <li>Tổ chức các policies theo Group</li> <li>Các policies mặc định</li> </ul>	1412653
4	Cài đặt VPD để có Fine-grained Access Control <ul style="list-style-type: none"> <li>Dùng DBMS_RLS Package và lệnh Create Context</li> <li>Dùng Giao diện Oracle policy manager</li> </ul>	1412006
5	Giới hạn phạm vi được sử dụng VPD (trong những schema nào)	1512454
6	Khái niệm Application Context <ul style="list-style-type: none"> <li>Các đặc trưng của Application Context và cách thiết lập</li> <li>Session-based/Local/Global Application context</li> <li>Cách làm Application Context cho Fine-grained Access Control</li> <li>Kiểm tra tác dụng policy: View V\$VPD_POLICY</li> </ul>	1512168, 1512454

7	Demo full	1512454, 1512168
---	-----------	------------------

## II. VIRTUAL PRIVATE DATABASE (VPD)

### A. KHÁI NIỆM VPD

- Virtual Private Database (Viết tắt là Oracle VPD) là một cơ chế AC của Oracle, công cụ kết hợp việc sử dụng công cụ điều khiển truy cập (FGAC – Fine-Grained Access Control) và ngữ cảnh ứng dụng (Application Context) từ đó cho phép người quản trị định nghĩa và áp dụng các chính sách về quản lý quyền truy cập tới mức dòng hoặc cột theo từng phiên làm việc.
- VPD cho phép định nghĩa các chính sách an toàn đến mức từng đối tượng (Table, View, Synonym) tương ứng với từng thao tác (SELECT, INSERT, UPDATE, DELETE). Khi người dùng trực tiếp hoặc gián tiếp truy cập vào đối tượng đã được thiết lập chính sách an toàn, hệ quản trị cơ sở dữ liệu sẽ tự động thay đổi câu lệnh SQL của người dùng bằng cách thêm vào câu lệnh SQL của người dùng mệnh đề WHERE hay còn gọi là vị ngữ (predicate) được trả về bởi hàm thực thi chính sách an toàn cho đối tượng đó. Việc thay đổi câu lệnh SQL này diễn ra trong suốt với người sử dụng cuối (người thực hiện câu lệnh SQL).
- Về mặt vật lý, khi áp dụng Oracle VPD người dùng cần có những thành phần như sau:
  - Application Context – Ngữ cảnh ứng dụng: Là nơi lưu trữ thông tin của ngữ cảnh cho ứng dụng.
  - PL/SQL Functions – Các hàm PL/SQL: Là các hàm thực thi các chính sách an toàn.
  - Security Policy – Các chính sách an toàn: Là các chính sách an toàn để áp dụng cho các đối tượng.
  - Chính sách an toàn được quản lý bởi các phương thức trong package DBMS\_RLS bao gồm các hàm:
    - ADD\_POLICY: thêm mới chính sách an toàn.
    - ENABLE\_POLICY: enable hoặc disable các chính sách an toàn.
    - REFRESH\_POLICY: Cập nhật lại hàm cho các chính sách an toàn
    - DROP\_POLICY: bỏ một chính sách an toàn

- Trước VPD:

USERNAME	NAME	SALARY	EMAIL
1 NV01	PHAN THANH SANG	1	PTSANG@GMAIL.COM
2 NV02	HAI HONG	10	HHONG@GMAIL.COM
3 NV03	BA DUY	100	BDUY@GMAIL.COM
4 NV04	NGUYEN THE HIEN	1000	NTHIEN@GMAIL.COM

ID_BENHNHAN	EMAIL_BACSI	BENH
1 BN0001	HHONG@GMAIL.COM	SOI THAN
2 BN0002	BDUY@GMAIL.COM	HIV
3 BN0003	NTHIEN@GMAIL.COM	UNG THU

- Sau VPD:

USERNAME	NAME	SALARY	EMAIL
1 NV01	PHAN THANH SANG	(null)	PTSANG@GMAIL.COM
2 NV02	HAI HONG	10	HHONG@GMAIL.COM
3 NV03	BA DUY	(null)	BDUY@GMAIL.COM
4 NV04	NGUYEN THE HIEN	(null)	NTHIEN@GMAIL.COM

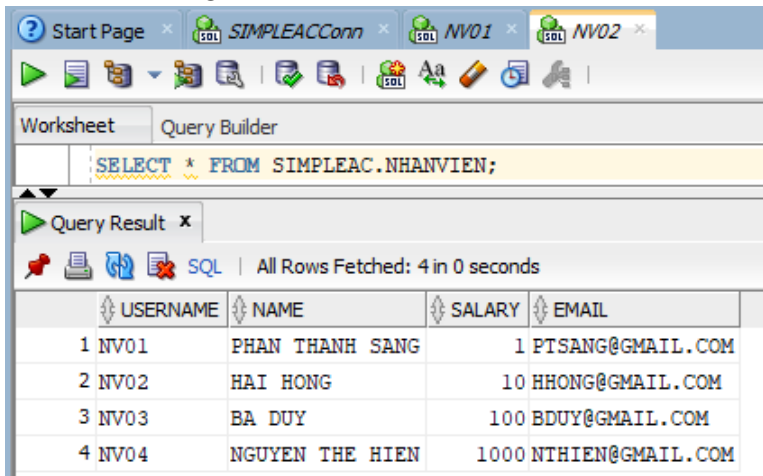
ID_BENHNHAN	EMAIL_BACSI	BENH
1 BN0001	HHONG@GMAIL.COM	SOI THAN



## B. COLUMN-LEVEL VPD VÀ COLUMN MASKING

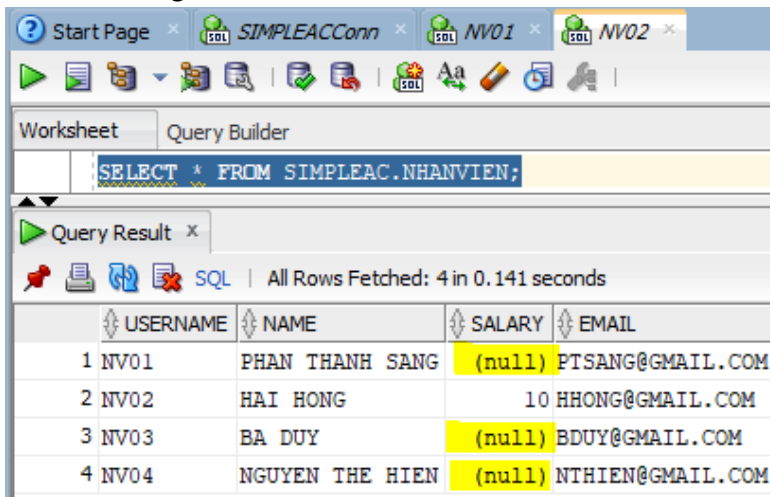
### 1. COLUMN LEVEL VPD

- Cung cấp các điều khiển truy cập chi tiết hơn trên dữ liệu.
- Gắn hàm chính sách trên cột cần hạn chế truy cập, thay vì gắn trên table hoặc view:
  - Mặc định, hiển thị tất cả những cột của chỉ các dòng thỏa chính sách.
  - Hoặc hiển thị tất cả những dòng dữ liệu, với các dòng mà người dùng không được phép truy cập tất cả các cột thì hiện null tại các cột cần giấu.
  - Tuy nhiên:
    - Điều khiển truy cập trên cột chỉ có thể ứng dụng cho câu 'select'.
    - Vị từ phải đơn giản.
- Trước khi dùng Column level:



	USERNAME	NAME	SALARY	EMAIL
1	NV01	PHAN THANH SANG	1	PTSANG@GMAIL.COM
2	NV02	HAI HONG	10	HHONG@GMAIL.COM
3	NV03	BA DUY	100	BDUY@GMAIL.COM
4	NV04	NGUYEN THE HIEN	1000	NTHIEN@GMAIL.COM

- Sau khi dùng Column level:



	USERNAME	NAME	SALARY	EMAIL
1	NV01	PHAN THANH SANG	(null)	PTSANG@GMAIL.COM
2	NV02	HAI HONG	10	HHONG@GMAIL.COM
3	NV03	BA DUY	(null)	BDUY@GMAIL.COM
4	NV04	NGUYEN THE HIEN	(null)	NTHIEN@GMAIL.COM

## 2. COLUMN MASKING

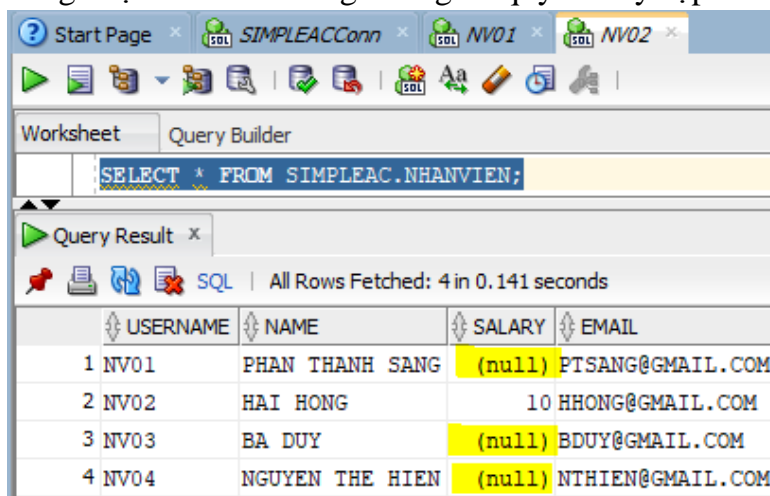
- Dùng để hiển thị các cột chứa dữ liệu nhạy cảm dưới dạng giá trị NULL.
- Nếu một truy vấn tham chiếu một cột chứa dữ liệu nhạy cảm (dữ liệu mà người dùng đó không được phép truy vấn), thì theo mặc định, VPD sẽ hạn chế số lượng dòng được trả về.
- Với COLUMN MASKING, tất cả các dòng hiển thị, ngay cả những dòng chứa những cột có dữ liệu nhạy cảm. Tuy nhiên, các cột nhạy cảm hiển thị dưới dạng giá trị NULL. Để áp dụng COLUMN MASKING, bạn có thể thay đổi tham số SEC\_RELEVANT\_COLS\_OPT của DBMS\_RLS.ADD\_POLICY procedure thành DBMS\_RLS.ALL\_ROWS.

```

=====
BEGIN
  DBMS_RLS.ADD_POLICY
  (
    OBJECT_SCHEMA      => 'SIMPLEAC',
    OBJECT_NAME        => 'NHANVIEN',
    POLICY_NAME        => 'VPD_NHANVIEN_POLICY',
    POLICY_FUNCTION     => 'VPD_NHANVIEN',
    SEC_RELEVANT_COLS   => 'SALARY',
    SEC_RELEVANT_COLS_OPT => DBMS_RLS.ALL_ROWS
  );
END;
/
=====

```

- Ví dụ COLUMN MASKING: những cột được bảo vệ biến thành null nếu người dùng chọn đến mà không có quyền truy cập.



The screenshot shows the SQL Developer interface with a query window open. The query is `SELECT * FROM SIMPLEAC.NHANVIEN;`. The query result is displayed in a table with 4 rows. The SALARY column is masked, showing NULL values for rows 1, 3, and 4, and a value of 10 for row 2.

	USERNAME	NAME	SALARY	EMAIL
1	NV01	PHAN THANH SANG	(null)	PTSANG@GMAIL.COM
2	NV02	HAI HONG	10	HHONG@GMAIL.COM
3	NV03	BA DUY	(null)	BDUY@GMAIL.COM
4	NV04	NGUYEN THE HIEN	(null)	NTHIEN@GMAIL.COM



## C. VPD SECURITY POLICY

### 1. DBMS\_RLS PROCEDURES

Procedure – thủ tục	Description – mô tả
<b>Tổ chức Policies riêng lẻ</b>	–
DBMS_RLS.ADD_POLICY	Thêm policy vào table, view, synonym
DBMS_RLS.ENABLE_POLICY	Cho phép (hay vô hiệu hóa) một policy trước đó mà ta đã thêm vào table, view, synonym
DBMS_RLS.ALTER_POLICY	Thay đổi liên kết hay không liên kết thuộc tính với policy
DBMS_RLS.REFRESH_POLICY	Vô hiệu hóa các cursor liên kết với policy, làm mới policy.
DBMS_RLS.DROP_POLICY	Xóa một policy ra khỏi table, view, synonym
<b>Tổ chức Policies theo group</b>	–
DBMS_RLS.CREATE_POLICY_GROUP	Tạo policy theo group
DBMS_RLS.ALTER_GROUPED_POLICY	Thay đổi nhóm policy
DBMS_RLS.DELETE_POLICY_GROUP	Xóa nhóm policy
DBMS_RLS.ADD_GROUPED_POLICY	Thêm một policy vào một nhóm policy xác định
DBMS_RLS.ENABLE_GROUPED_POLICY	Cho phép một policy thuộc trong nhóm policy xác định
DBMS_RLS.REFRESH_GROUPED_POLICY	Phân tích lại câu lệnh SQL liên kết với policy đã được làm mới.
DBMS_RLS.DISABLE_GROUPED_POLICY	Vô hiệu hóa một policy thuộc trong nhóm policy xác định
DBMS_RLS.DROP_GROUPED_POLICY	Xóa một policy thuộc trong nhóm policy xác định (xóa một thành viên của nhóm policy).
<b>Tổ chức ngữ cảnh của ứng dụng</b>	–
DBMS_RLS.ADD_POLICY_CONTEXT	Thêm ngữ cảnh vào ứng dụng đang hoạt động
DBMS_RLS.DROP_POLICY_CONTEXT	Xóa một ngữ cảnh của ứng dụng

### 2. ÁP DỤNG CÁC POLICIES TRÊN TABLE, VIEW, SYNONYM

- Bạn có thể bảo vệ các đối tượng dữ liệu bằng cách phân quyền thêm xóa, cập nhật, chế độ xem dữ liệu... đối với người dùng khác nhau, hạn chế mất mát thông tin tối đa nhất có thể. Vì vậy, bạn cần thiết kế và sử dụng các chính sách bảo mật để hạn chế quyền truy cập vào các table, view, synonym.

- Ví dụ 1: Quản lý hệ thống gọi điện thoại mời khách hàng tham gia bảo hiểm của công ty bảo hiểm nhân thọ, mỗi nhân viên sẽ có danh sách các số điện thoại cần gọi trong ngày. Nếu nhiều nhân viên cùng gọi cho 1 khách hàng sẽ gây khó chịu, giống như bị quấy rối vậy, nên danh sách số điện thoại nhất thiết phải khác nhau. Mỗi nhân viên sẽ nhìn thấy table danh sách khác nhau, và có thao tác xử lý chỉ là xem thông tin khách hàng, gọi điện thoại, và cập nhật thông tin, không có quyền xóa hay thêm khách hàng; để đảm bảo an toàn dữ liệu. Nhân viên admin có thể xem toàn bộ table danh sách khách hàng và có mọi quyền truy cập.
- Minh họa: table KHACHHANG được view khác nhau giữa các nhân viên.
  - Table KHACHHANG: view của admin.

ID	Tên	Ngày sinh	Số điện thoại	Loại bảo hiểm
512	Ánh	28/09/1990	0912345678	C
513	Như	08/01/1995	0987634567	-
514	Bảo	14/09/1987	0312567891	-
515	Cường	05/09/1997	0124578906	-
516	Hoàng	06/12/1996	0876523417	-
517	Trung	10/01/1991	0956789345	-

- Xuất view cho các nhân viên với các thông tin ID, tên, số điện thoại, của khách hàng có loại bảo hiểm bằng NULL
- View của nhân viên A:

ID	Tên	Số điện thoại
513	Như	0987634567
515	Cường	0124578906
517	Trung	0956789345

View của nhân viên B:

ID	Tên	Số điện thoại
514	Bảo	0312567891
516	Hoàng	0876523417

- Giải pháp: dùng VPD, tạo vào áp dụng policies trên table KHACHHANG.
- Ưu điểm:
  - Giả sử trong lần thay đổi nhân sự, nhân viên A được làm admin, thì chỉ cần thay đổi chính sách bảo mật đối với tài khoản của nhân viên A, mà không cần chỉ sửa các view khác => đơn giản, nhanh chóng.
  - Bảo mật vì áp dụng chính sách bảo mật trên chính dữ liệu gốc. (Nếu áp dụng trên bản sao thì dữ liệu gốc có thể không đảm bảo hạn chế mất mát thông tin)
- Nguyên lý hoạt động:
  - Truy cập đến table, view hay synonym có khai báo chính sách VPD

- Tạo hàm chính sách, hàm chính sách trả về một vị từ (predicate) dựa vào nội dung của dữ liệu hay ngữ cảnh ứng dụng
- Demo:
  - Admin có thể truy cập tất cả các đối tượng dữ liệu, các nhân viên chỉ được truy cập dữ liệu của chính họ.
  - Công ty có bảng dữ liệu khách hàng.

KHACHHANG (OWNER VARCHAR2(20), DATA VARCHAR2(20));

- VARCHAR2(n) là kiểu chuỗi ký tự unicode có độ dài bất kỳ đến n ký tự, tối đa 4000 byte.
- Tạo hàm chính sách:

```
CREATE FUNCTION SEC_POLICY (SCHEMA VARCHAR2, OBJECT VARCHAR2)
RETURN VARCHAR2
AS
    USER VARCHAR2(100);
BEGIN
    IF (SYS_CONTEXT ('USERENV', 'ISDBA')) THEN
        RETURN '';
    ELSE
        USER := SYS_CONTEXT('USERENV','SESSION_USER');
        RETURN 'OWNER =' || user;
    END IF;
END;
```

- Hàm SYS\_CONTEXT để truy xuất thông tin về môi trường của Oracle.
- Cú pháp: SYS\_CONTEXT (namespace, parameter [, length] )
- namespace: ở đây là 'USERENV' dùng để truy thông tin phiên bản của Oracle hiện tại.

- parameter: ở đây là 'ISDBA' dùng để trả về TRUE nếu người dùng có đặt quyền DBA - Database Administrator, nếu không thì trả về FALSE. Tham số 'SESSION\_USER' dùng để chỉ tên người dùng cơ sở dữ liệu của người dùng dùng để đăng nhập.
- length: là độ dài của giá trị trả về tính bằng byte. Nếu tham số này bị bỏ qua, hay có nhập nhưng không hợp lệ thì mặc định là 256 byte.
- Thêm hàm chính sách vào table KHACHHANG:

```
BEGIN  
  
DBMS_RLS.ADD_POLICY  
(  
  
    OBJECT_SCHEMA => 'schema',  
    OBJECT_NAME   => 'KHACHHANG',  
    POLICY_NAME   => 'KH_policy',  
    FUNCTION_SCHEMA => 'pkg',  
    POLICY_FUNCTION => 'SEC_FUCTION',  
    STATEMENT_TYPES => 'SELECT, UPDATE, INSERT',  
    UPDATE_CHECK   => TRUE  
);  
  
END;
```

- Đầu tiên, để thêm một policy vào table, view, synonym, bạn có thể sử dụng thủ tục DBMS\_RLS.ADD\_POLICY trong DBMS\_RLS PL/SQL package.
- Cần phải chỉ định rõ table, view hay synonym nào mà bạn muốn thêm policy và xác định tên của policy đó. Ngoài ra, cần xác định các thông tin khác, như là các câu lệnh xử lý policy như SELECT, INSERT, UPDATE, DELETE, CREATE INDEX, hay ALTER INDEX.

- Có thể thêm các câu lệnh xử lý cho policy như SELECT, INSERT, UPDATE, DELETE, CREATE INDEX, hay ALTER INDEX bằng tham số 'STATEMENT\_TYPES'.
- Thêm hàm chính sách SEC\_POLICY trong scheme pkg đã được tạo ở trên vào đối tượng dữ liệu table là bảng 'schema.KHACHHANG' và đặt tên của chính sách là KH\_policy, sử dụng một số câu lệnh xử lý như select, update, insert.
- Các parameter – tham số và ý nghĩa:

	Kiểu dữ liệu	NULL	Mô tả
<b>OBJECT_OWNER</b>	VARCHAR2(30)	NOT NULL	Owner của table, view, hay synonym (thuộc quyền sở hữu)
<b>OBJECT_NAME</b>	VARCHAR2(30)	NOT NULL	Tên của table, view, hay synonym
<b>POLICY_GROUP</b>	VARCHAR2(30)	NOT NULL	Tên của nhóm policy
<b>POLICY_NAME</b>	VARCHAR2(30)	NOT NULL	Tên của policy
<b>PF_OWNER</b>	VARCHAR2(30)	NOT NULL	Owner hàm của policy
<b>PACKAGE</b>	VARCHAR2(30)		Tên của package chứa hàm của policy
<b>FUNCTION</b>	VARCHAR2(30)	NOT NULL	Tên hàm của policy
<b>SEL</b>	VARCHAR2(3)		Cho biết policy có được ứng dụng cho các truy vấn trên đối tượng hay không (yes – no)
<b>INS</b>	VARCHAR2(3)		Cho biết policy có được ứng dụng cho các câu lệnh INSERT trên đối tượng hay không (yes – no)
<b>UPD</b>	VARCHAR2(3)		Cho biết policy có được ứng dụng cho các câu lệnh UPDATE trên đối tượng hay không (yes – no)
<b>DEL</b>	VARCHAR2(3)		Cho biết policy có được ứng dụng cho các câu lệnh DELETE trên đối tượng hay không (yes – no)

<b>IDX</b>	VARCHAR2(3)		Cho biết policy có được thực thi để duy trì các INDEX trên đối tượng hay không (yes – no)
<b>CHK_OPTION</b>	VARCHAR2(3)		Cho biết có được kiểm tra option khi thực thi policy hay không (yes – no)
<b>ENABLE</b>	VARCHAR2(3)		Cho biết policy đang ở trạng thái cho phép – enable hay vô hiệu hóa – disable (yes – no)
<b>STATIC_POLICY</b>	VARCHAR2(3)		Cho biết policy có đang ở trạng thái static hay không (yes – no)
<b>POLICY_TYPE</b>	VARCHAR2(24)		Policy type: <ul style="list-style-type: none"> <li>• STATIC</li> <li>• SHARED_STATIC</li> <li>• CONTEXT_SENSITIVE</li> <li>• SHARED_CONTEXT_SENSITIVE</li> <li>• DYNAMIC</li> </ul>
<b>LONG_PREDICATE</b>	VARCHAR2(3)		Cho biết hàm của policy có giá trị trả về tối đa là 32KB đúng hay không (yes – no); nếu không thì mặc định là 4000 byte

- Nhân viên A truy cập vào table KHACHHANG:

--Câu lệnh được nhân viên execute:

```
SELECT *FROM KHACHHANG;
```

--Câu lệnh sau khi quan kiểm tra VPD:

```
SELECT *FROM KHACHHANG WHERE OWNER = 'A';
```

- Lưu ý:
  - Ta chỉ có thể xác định policy nào đang có trong table, và không thể xác định ngược lại là xác định table nào mà trong có policy đã được thêm vào. Hay nói cách khác, dựa vào table, ta có thể xác định các policy nào có trong table, và không thể dựa vào policy, để biết policy này đã được thêm vào table nào.



### 3. TỔ CHỨC POLICIES THEO GROUP

- Tổ chức các chính sách theo nhóm là hành động nhóm nhiều chính sách bảo mật lại với nhau, có đặt tên cho nhóm đó và áp dụng cho một ứng dụng cụ thể. Nhóm chính sách là tập hợp các chính sách bảo mật của một ứng dụng. Dựa vào ngữ cảnh của ứng dụng mà chỉ ra nhóm chính sách nào có hiệu lực. Khi một nhóm chính sách có hiệu lực, nó sẽ thực thi tất cả các chính sách liên quan thuộc về nhóm chính sách đó.
- Ví dụ 2: Công ty A có 2 chi nhánh là NS và TC. Công ty ABNC lưu trữ bảng phụ cấp (PhuCap) của cả 2 chi nhánh. Hai chi nhánh có 2 ứng dụng khác nhau là Nhân Sự và Tài Chính với 2 chính sách bảo mật khác nhau. Ứng dụng Nhân Sự phân quyền cho người dùng dựa trên cấp bậc bộ phận. Ứng dụng Tài Chính phân quyền cho người dùng dựa theo bảng xếp hạng trong công ty. Việc cần tích hợp 2 chính sách này vào bảng PHUCAP, đòi hỏi cần phải xây dựng chính sách chung của 2 chi nhánh, là việc không khả thi.
- Bằng cách xác định ngữ cảnh của ứng dụng để xây dựng một bộ chính sách cụ thể cho các đối tượng dữ liệu, với mỗi ứng dụng có thể sử dụng bộ chính sách bảo mật riêng. Để làm được điều này bạn cần tổ chức các chính sách bảo mật thành nhóm. Dựa vào ngữ cảnh của ứng dụng mà chỉ ra nhóm chính sách nào có hiệu lực. Khi một nhóm chính sách có hiệu lực, nó sẽ thực thi tất cả các chính sách liên quan thuộc về nhóm chính sách đó.
- Tạo nhóm chính sách:
  - Thủ tục DBMS\_RLS.ADD\_GROUPED\_POLICY thêm một chính sách vào một nhóm chính sách.
  - Để chỉ định ra nhóm chính sách nào có hiệu lực, cần thêm ngữ cảnh của ứng dụng bằng cách sử dụng thủ tục DBMS\_RLS.ADD\_POLICY\_CONTEXT. Nếu trả về nhóm chính sách không xác định thì đã xảy ra lỗi. Khi xảy ra lỗi, cơ sở dữ liệu Oracle sẽ chạy tất cả các chính sách. Vì vậy bước này khá quan trọng, bạn không nên bỏ qua bước thêm ngữ cảnh cho ứng dụng này, tránh trường hợp mọi chính sách được áp dụng.
  - Bạn cũng có thể thêm nhiều ngữ cảnh của ứng dụng cho cùng một table, view, synonym. Khi đó, các bối cảnh cần được xử lý riêng lẻ. Điều này cho phép bạn cài đặt nhiều bộ chính sách cùng được áp dụng.
  - Ví dụ 3: Mô tả như ví dụ 1, theo như chiến thuật thì khách hàng nam sẽ cho nhân viên nữ gọi và ngược lại, để tăng doanh số của công ty, và công ty thì có nhiều nhân viên nữ. Người dùng ở đây là nhân viên nữ G, ở đây có 2 ngữ cảnh là khách hàng phải là người có giới tính là nam và danh sách

khách hàng từ khoảng X->Y. Dựa vào ngữ cảnh ứng dụng trên mà xác định nhóm chính sách, hay các nhóm chính sách nào được thực thi.

- Thủ tục DBMS\_RLS.CREATE\_POLICY\_GROUP: Tạo một nhóm chính sách.
  - Cú pháp:

```
DBMS_RLS.CREATE_POLICY_GROUP (
```

```
    object_schema  VARCHAR2,
```

```
    object_name    VARCHAR2,
```

```
    policy_group   VARCHAR2);
```

- Ví dụ: Tạo một nhóm chính sách có tên là Grp\_policy thuộc bảng bảng schema.KHACHHANG

```
DBMS_RLS.CREATE_POLICY_GROUP (
```

```
    'schema', 'KHACHHANG', 'Grp_policy');
```

- Thủ tục DBMS\_RLS.DELETE\_POLICY\_GROUP: Xóa một nhóm chính sách.
  - Cú pháp:

```
DBMS_RLS.DELETE_POLICY_GROUP (
```

```
    object_schema  VARCHAR2,
```

```
    object_name    VARCHAR2,
```

```
    policy_group   VARCHAR2);
```

- Ví dụ: Xóa một nhóm chính sách có tên là Grp\_policy thuộc bảng bảng schema.KHACHHANG

```
DBMS_RLS.DELETE_POLICY_GROUP (
```

```
    'schema', 'KHACHHANG', 'Grp_policy');
```

- Thủ tục DBMS\_RLS.ADD\_GROUPED\_POLICY: Thêm một chính sách vào một nhóm chính sách.
  - Cú pháp:

```
DBMS_RLS.ADD_GROUPED_POLICY(
    object_schema    VARCHAR2,
    object_name      VARCHAR2,
    policy_group     VARCHAR2,
    policy_name      VARCHAR2,
    function_schema  VARCHAR2,
    policy_function   VARCHAR2,
    statement_types  VARCHAR2,
    update_check     BOOLEAN,
    enabled          BOOLEAN,
    static_policy    IN BOOLEAN FALSE,
    policy_type      IN BINARY_INTEGER NULL,
    long_predicate   IN BOOLEAN FALSE,
    sec_relevant_cols IN VARCHAR2,
    sec_relevant_cols_opt IN BINARY_INTEGER NULL);
```

- Ví dụ: Tên của chính sách muốn thêm vào là “policy\_A”, với hàm chính sách “pkg.SEC\_FUCTION”

```
DBMS_RLS.ADD_GROUPED_POLICY(
    'schema', 'KHACHHANG', 'Grp_policy', 'policy_A', 'pkg',
    SEC_FUCTION');
```

- Thủ tục DBMS\_RLS.ENABLE\_GROUPED\_POLICY: Cho phép (hay vô hiệu hóa) một chính sách có trong một nhóm chính sách xác định.
  - Cú pháp:

```
DBMS_RLS.ENABLE_GROUPED_POLICY (
    object_schema VARCHAR2,
    object_name   VARCHAR2,
```

```
group_name    VARCHAR2,  
policy_name   VARCHAR2,  
enable        BOOLEAN);
```

- Ví dụ: Cho phép chính sách policy\_A thuộc nhóm chính sách Grp\_policy trong bảng schema.KHACHHANG

```
DBMS_RLS.ENABLE_GROUPED_POLICY (  
    'schema', 'KHACHHANG', 'Grp_policy', 'policy_A', TRUE);
```

- Thủ tục DBMS\_RLS.REFRESH\_GROUPED\_POLICY: Làm mới một chính sách trong một nhóm xác định
  - Cú pháp:

```
DBMS_RLS.REFRESH_GROUPED_POLICY (  
    object_schema VARCHAR2,  
    object_name    VARCHAR2,  
    group_name     VARCHAR2,  
    policy_name    VARCHAR2);
```

- Ví dụ: Làm mới chính sách policy\_A thuộc nhóm chính sách Grp\_policy trong bảng schema.KHACHHANG

```
DBMS_RLS.REFRESH_GROUPED_POLICY (  
    'schema', 'KHACHHANG', 'Grp_policy', 'policy_A');
```

- Thủ tục DBMS\_RLS.DISABLE\_GROUPED\_POLICY: Vô hiệu hóa một chính sách trong một nhóm xác định.
  - Cú pháp:

```
DBMS_RLS.DISABLE_GROUPED_POLICY (  
    object_schema VARCHAR2,  
    object_name    VARCHAR2,
```

```
group_name    VARCHAR2,  
policy_name   VARCHAR2);
```

- Ví dụ: Vô hiệu hóa chính sách policy\_A thuộc nhóm chính sách Grp\_policy trong bảng schema.KHACHHANG

```
DBMS_RLS.DISABLE _GROUPED_POLICY (  
    'schema', 'KHACHHANG', 'Grp_policy', 'policy_A');
```

- Thủ tục DBMS\_RLS.DROP\_GROUPED\_POLICY: Loại bỏ một chính sách trong một nhóm xác định.
  - Cú pháp:

```
DBMS_RLS.DROP_GROUPED_POLICY (  
    object_schema  VARCHAR2,  
    object_name    VARCHAR2,  
    policy_group   VARCHAR2,  
    policy_name    VARCHAR2);
```

- Ví dụ: Cho phép chính sách policy\_A thuộc nhóm chính sách Grp\_policy trong bảng schema.KHACHHANG

```
DBMS_RLS.DROP _GROUPED_POLICY (  
    'schema', 'KHACHHANG', 'Grp_policy', 'policy_A');
```

---

#### 4. CÁC POLICIES MẶC ĐỊNH

- Hàm SYS\_CONTEXT: truy xuất thông tin về môi trường của oracle
  - Cú pháp: Hàm trả về giá trị là chuỗi

```
SYS_CONTEXT(namespace, parameter [, length] )
```

- *namespace*: Một Oracle namespace đã được tạo trước đó. Nếu Oracle namespace là 'USERENV' dùng để truy thông tin phiên bản của Oracle hiện tại.
- *length*: là độ dài của giá trị trả về tính bằng byte. Nếu tham số này bị bỏ qua, hay có nhập nhưng không hợp lệ thì mặc định là 256 byte.
- *parameter*: Tham số đã được định nghĩa sẵn; một số tham số hợp lệ của namespace là 'USERENV' (không phải tất cả các tham số đều hợp lệ trong tất cả phiên bản của Oracle)
- Ví dụ: Lọc thông tin trong bảng OE.ORDERS theo người dùng.  
SESSION\_USER dùng để chỉ tên người dùng cơ sở dữ liệu của người dùng dùng để đăng nhập.

```
SELECT * FROM OE.ORDERS  
WHERE SALES_REP_ID =  
SYS_CONTEXT('USERENV','SESSION_USER');
```



## D. CÀI ĐẶT VPD ĐỂ CÓ FINE-GAINED ACCESS CONTROL

- Table

	E_ID	USERNAME	E_NAME	E_SALARY
1	1	NV01	GIAM DOC	100000000
2	2	NV02	TRUONG PHONG	20000000
3	3	NV03	NHAN VIEN 3	3000000
4	4	NV04	NHAN VIEN 4	4000000
5	5	NV05	NHAN VIEN 5	5000000

- Tạo và gán quyền cho SIMPLEVPD là người thực thi các policy

1	CREATE USER SIMPLEVPD IDENTIFIED BY 123456;
2	GRANT DBA TO SIMPLEVPD;
3	GRANT CREATE SESSION, CREATE ANY CONTEXT, CREATE PROCEDURE,
	CREATE TRIGGER, SIMPLEVPDISTER DATABASE TRIGGER TO SIMPLEVPD;
4	GRANT EXECUTE ON DBMS_SESSION TO SIMPLEVPD;
5	GRANT EXECUTE ON DBMS_RLS TO SIMPLEVPD;

- Tạo các user

1	CREATE USER NV01 IDENTIFIED BY 123456;	GRANT CONNECT TO NV01;
2	CREATE USER NV02 IDENTIFIED BY 123456;	GRANT CONNECT TO NV02;
3	CREATE USER NV03 IDENTIFIED BY 123456;	GRANT CONNECT TO NV03;
4	CREATE USER NV04 IDENTIFIED BY 123456;	GRANT CONNECT TO NV04;
5	CREATE USER NV05 IDENTIFIED BY 123456;	GRANT CONNECT TO NV05;

- Tạo function có chuỗi trả về là ...

```
CREATE OR REPLACE FUNCTION VPD_EMPLOYEES(SCHEMA
VARCHAR2,OBJECT VARCHAR2)
RETURN VARCHAR2
AS
```

```
USER VARCHAR2(100);  
BEGIN  
    USER := SYS_CONTEXT('USERENV', 'SESSION_USER');  
    RETURN 'USERNAME =' || ' USER';  
END;  
/
```

- Thực thi chính sách

```
BEGIN  
    DBMS_RLS.ADD_POLICY  
    (  
        OBJECT_SCHEMA =>'SIMPLEVPD',  
        OBJECT_NAME    =>'EMPLOYEES',  
        POLICY_NAME     =>'VPD_EMPLOYEES_POLICY',  
        POLICY_FUNCTION =>'VPD_EMPLOYEES',  
        STATEMENT_TYPES =>'SELECT, DELETE'  
    );  
END;  
/
```

- Sau khi thực thi chính sách người thực thi (SIMPLEVPD) không thể thấy được dữ liệu

Start Page x NV01 x SIMPLEVPD x NV05 x NV04 x NV03 x NV02 x

0.002 seconds

Worksheet Query Builder

```
SELECT * FROM SIMPLEVPD.EMPLOYEES;
```

```
CREATE OR REPLACE FUNCTION VPD_EMPLOYEES (SCHEMA VARCHAR2, OBJECT VARCHAR2)
RETURN VARCHAR2
AS
  USER VARCHAR2(100);
BEGIN
  USER := SYS_CONTEXT('USERENV', 'SESSION_USER');
  RETURN 'USERNAME = ' || ' USER';
END;
/
```

```
BEGIN
  DBMS_RLS.ADD_POLICY
  (
    OBJECT_SCHEMA => 'SIMPLEVPD',
    OBJECT_NAME   => 'EMPLOYEES',
    POLICY_NAME    => 'VPD_EMPLOYEES_POLICY',
    POLICY_FUNCTION => 'VPD_EMPLOYEES',
    STATEMENT_TYPES => 'SELECT, DELETE'
  );
END;
/
```

Script Output x

Task completed in 0.002 seconds

no rows selected

- Nhân viên chỉ xem được SALARY và E\_NAME của chính họ

Start Page x NV01 x SIMPLEVPD x NV05 x NV04 x NV03 x NV02 x EMPLOYEES x

0.003 seconds

Worksheet Query Builder

```
SELECT * FROM SIMPLEVPD.EMPLOYEES;
```

Script Output x

Task completed in 0.003 seconds

E_ID	USERNAME	E_NAME	E_SALARY
5	NV05	NHAN VIEN 5	5000000

- DROP chính sách cũ

DROP FUNCTION VPD\_EMPLOYEES;

```
BEGIN
  DBMS_RLS.DROP_POLICY (
    OBJECT_SCHEMA=> 'SIMPLEVPD',
    OBJECT_NAME => 'EMPLOYEES',
    POLICY_NAME => 'VPD_EMPLOYEES_POLICY'
  );
END;
/
```

- Tạo chính sách mới cho phép user có thể xem thông tin các user các cho phép (E\_NAME ), không cho xem các thông tin ( E\_SALARY ).

```
BEGIN
  DBMS_RLS.ADD_POLICY
  (
    OBJECT_SCHEMA    =>'SIMPLEVPD',
    OBJECT_NAME      =>'EMPLOYEES',
    POLICY_NAME       =>'VPD_EMPLOYEES_POLICY',
    POLICY_FUNCTION   => 'VPD_EMPLOYEES',
    SEC_RELEVANT_COLS => 'E_SALARY',
    STATEMENT_TYPES   => 'SELECT',
    SEC_RELEVANT_COLS_OPT=> DBMS_RLS.ALL_ROWS
  );
END;
/
```

- Sau khi thực thi nhân viên xem được thông tin các user khác. Nhưng chỉ xem được E\_SALARY của chính họ.

Start Page × NV01 × SIMPLEVPD × NV05 × NV04 × NV03 × NV02 × EMPLOYEES ×

0.004 seconds

Worksheet Query Builder

```
SELECT * FROM SIMPLEVPD.EMPLOYEES;
```

Script Output ×

Task completed in 0.004 seconds

E_ID	USERNAME	E_NAME	E_SALARY
1	NV01	GIAM D?C	
2	NV02	TRU?NG PHONG	
3	NV03	NHAN VIEN 3	
4	NV04	NHAN VIEN 4	
5	NV05	NHAN VIEN 5	5000000

Start Page × NV01 × SIMPLEVPD × NV05 × NV04 × NV03 × NV02 × EMPLOYEES ×

0.004 seconds

Worksheet Query Builder

```
SELECT * FROM SIMPLEVPD.EMPLOYEES;
```

Script Output ×

Task completed in 0.004 seconds

E_ID	USERNAME	E_NAME	E_SALARY
1	NV01	GIAM D?C	100000000
2	NV02	TRU?NG PHONG	
3	NV03	NHAN VIEN 3	
4	NV04	NHAN VIEN 4	
5	NV05	NHAN VIEN 5	

## E. GIỚI HẠN PHẠM VI SỬ DỤNG ĐƯỢC VPD (NHỮNG SCHEMA NÀO)

- Khi export data, chính sách VPD không được thi hành khi export data trực tiếp từ đường dẫn. Khi xuất dữ liệu trực tiếp từ một đường dẫn, Oracle Database đọc dữ liệu từ đĩa vào bộ đệm và chuyển dữ liệu trực tiếp đi.
- Bạn không thể áp dụng chính sách VPD lên đối tượng trong SYS schema. Người dùng SYS và những người tạo được connect as SYSDBA thì chính sách VPD không tác động lên hành động của họ.
  - Mặc dù vậy, bạn có thể Audit người dùng SYS để xem những hành động của họ thực hiện lên database và lưu trong một chỗ an toàn trong hệ điều hành.

```
Enter user-name: sys/123456 as sysdba
Connected to:
Oracle Database 12c Enterprise Edition Release 12.2.0.1.0 - 64bit Production
SQL> grant dba to kda;
Grant succeeded.
SQL> conn kda/123456
Connected.
SQL> select * from simpleac.benhnhhan;
no rows selected
```

```
SQL> conn kda/123456 AS SYSDBA
Connected.
SQL> select * from simpleac.benhnhhan;
```

ID_BENHNHAN	EMAIL_BACSI
BENH	
BN0001	HHONG@GMAIL.COM
SOI THAN	
BN0002	BDUY@GMAIL.COM
HIV	
BN0003	NTHIEN@GMAIL.COM
UNG THU	



- Những người dùng được cấp quyền EXEMPT ACCESS POLICY (GRANT EXEMPT ACCESS POLICY TO <USERNAME or ROLE>) dù là qua trực tiếp cấp hay trong Role cấp có đều vượt qua được VPD. Quyền này thường dùng để cấp cho các người dùng quản trị chẳng hạn như cài đặt, đặt dữ liệu và xuất dữ liệu không phải thuộc sys schema.

```
SQL> conn kda/123456
Connected.
SQL> select * from simpleac.benhnhhan;
no rows selected

SQL> conn sys/123456 as sysdba
Connected.
SQL> grant EXEMPT ACCESS POLICY to kda;

Grant succeeded.

SQL> conn kda/123456
Connected.
SQL> select * from simpleac.benhnhhan;
```

ID_BENHNHAN	EMAIL_BACSI
BENH	
BN0001 SOI THAN	HHONG@GMAIL.COM
BN0002 HIV	BDUY@GMAIL.COM
BN0003 UNG THU	NTHIEN@GMAIL.COM

### III. KHÁI NIỆM APPLICATION CONTEXT

#### A. CÁC ĐẶC TRƯNG CỦA APPLICATION CONTEXT VÀ CÁCH THIẾT LẬP

##### 1. KHÁI NIỆM

- Là một phần của Cơ sở dữ liệu riêng ảo (VPD) trong oracle, là tập hợp các cặp name-value (tên/thuộc tính – giá trị) (một mảng kết hợp) được lưu trữ trong bộ nhớ (UGA hoặc SGA).
- NCUD gồm 2 phần là name và value:
  - Name: Là tên/thuộc tính của tập thuộc tính liên kết với value. VD: Ta có AC empno\_ctx và ID nhân viên từ bảng HR.EMPLOYEES, thì nó sẽ có tên là employee\_id.
  - Value: là giá trị của một thuộc tính. VD: nếu AC empno\_ctx, nếu bạn lấy ID nhân viên từ bảng HR.EMPLOYEES, thì bạn có thể tạo một giá trị gọi là emp\_id để đặt giá trị cho ID.

##### 2. CÁC ĐẶC TRƯNG

- NCUD thường nhóm các thuộc tính có liên quan lại với nhau thành một nhóm và được truy cập theo nhãn (namespace) hay tên của nó. => Giúp việc truy xuất các giá trị nhanh hơn.
- Thông thường trong NCUD thường chứa các thuộc tính như tên người dùng, tổ chức, quy tắc hay tiêu đề và các chính sách bảo mật vẫn có thể tham chiếu đến các thuộc tính này khi người dùng đang đăng nhập. Trong các tài liệu bảo mật thường hay sử dụng NCUD.
- Ngăn cảnh ứng dụng được xem như một biến toàn cục chứa thông tin được truy cập trong cơ sở dữ liệu.
- Phần lớn các ứng dụng chứa thông tin có cài đặt NCUD. VD: Trong trường hợp doanh nghiệp nhận đơn hàng có sử dụng các bảng OrderNumber và CustomerNumber, thì doanh nghiệp có thể sử dụng các giá trị trong các cột này làm thuộc tính để bảo mật để hạn chế quyền truy cập của khách hàng đối với các đơn hàng của mình bằng cách sử dụng ID của khách hàng đó.

### 3. LỢI ÍCH

- Kiểm soát việc truy cập chi tiết. VD: Chính sách Cơ sở dữ liệu riêng ảo của Oracle
- Bảo mật danh tính người dùng trên các môi trường khác nhau
- Giúp cho việc bảo mật trong các ứng dụng của bạn trở nên mạnh mẽ hơn vì NCUD được kiểm soát bởi một quy trình đáng tin cậy chứ không phải được kiểm soát bởi người dùng.
- Được sử dụng như một bộ đệm dữ liệu an toàn cho các thuộc tính của một ứng dụng: để kiểm soát chi tiết hoặc để sử dụng trong các câu lệnh hoặc vòng lặp có điều kiện trong PL/SQL.

## B. SESSION-BASED/LOCAL/GLOBAL APPLICATION CONTEXT

### 1. CLIENT SESSION-BASED APPLICATION CONTEXT (NGŨ CẢNH ỨNG DỤNG DỰA TRÊN LẦN ĐĂNG NHẬP CỦA KHÁCH HÀNG)

- Là loại ngữ cảnh ứng dụng các hàm trong OIC (Giao diện cuộc gọi của Oracle – Oracle call interface) ở bên máy của người dùng để đặt các dữ liệu trên lần đăng nhập này và thực hiện các kiểm tra cần thiết để hạn chế quyền truy cập của người dùng. Loại ngữ cảnh này được lưu trữ trong bộ nhớ SGA (System Global Area).

### 2. DATABASE SESSION-BASED APPLICATION CONTEXTS (NGŨ CẢNH ỨNG DỤNG DỰA TRÊN LẦN LÀM VIỆC)

- Là loại NCUD mà cơ sở dữ liệu oracle đặt các giá trị khi người dùng đăng nhập và sau khi người dùng thoát khỏi hệ thống thì CSDL sẽ tự động xóa các giá trị này khỏi bộ đệm của hệ thống. Nếu người dùng thoát một cách đột xuất (VD như mất điện) thì hệ thống sẽ tự xóa giá trị khỏi bộ đệm mà không cần bạn phải xóa.

NCUD này gồm 3 loại:

- Initialized locally (Khởi tạo cục bộ): NCUD được tạo cục bộ, dựa vào lần đăng nhập của người dùng.
- Initialized externally (Khởi tạo từ bên ngoài): NCUD được tạo từ OIC, theo thứ tự hàng đợi hoặc liên kết với cơ sở dữ liệu của người dùng đang kết nối.
- Initialized globally (Khởi tạo chung): NCUD sử dụng các thuộc tính và giá trị từ một thư mục trung lập như LDAP.

### 3. GLOBAL APPLICATION CONTEXT (NGŨ CẢNH ỨNG DỤNG CHUNG)

- Là loại NCUD có thể truy cập bởi tất cả người dùng (được sử dụng cho các hệ thống có mô hình không theo lần làm việc như hệ thống trung cấp của mô hình 3 lớp) và được lưu trữ trong bộ nhớ SGA. Có 3 cách để sử dụng loại NCUD này:
  - Có người dùng phải chuyển CSDL từ ứng dụng này sang ứng dụng khác. Ứng dụng thứ 2 mà người dùng di chuyển có các yêu cầu đăng nhập khác với ứng dụng đầu tiên.
  - Phải xác thực người dùng cơ sở dữ liệu trong khi người đó không biết đến cơ sở dữ liệu này. VD: Các trang bán hàng online thường cho người dùng xem, đặt hàng của mình mà không cần yêu cầu khách hàng phải tạo tài khoản hoặc đăng nhập và cơ sở dữ liệu của mình
  - Cần chia sẻ các giá trị chung cho tất cả các người dùng trong cơ sở dữ liệu. VD: thông báo của công ty mà tất cả các nhân viên trong công ty có đều có thể xem được.

## C. CÁCH DÙNG APPLICATION CONTEXT CHO FINE-GAINED ACCESS CONTROL

### DBMS\_SESSION.SET\_CONTEXT

```
DBMS_SESSION.SET_CONTEXT
(
    namespace VARCHAR2,
    attribute  VARCHAR2,
    value      VARCHAR2,
    username   VARCHAR2,
    client_id  VARCHAR2
);
```

Tham số	Ghi chú
<b>Namespace</b>	Namespace của ngữ cảnh ứng dụng được đặt, tối đa 30 byte
<b>Attribute</b>	Thuộc tính của ngữ cảnh ứng dụng, tối đa 30 byte
<b>Value</b>	Giá trị của ngữ cảnh ứng dụng, giới hạn 4 Kilobyte
<b>Username</b>	Tên người dùng CSDL của ngữ cảnh ứng dụng. Mặc định: NULL
<b>Client_id</b>	Giá trị cụ thể của id máy khách của ngữ cảnh ứng dụng. Tối đa 64 byte. Mặc định: NULL

Ảnh hưởng của thuộc tính client\_id và username:

GIÁ TRỊ CẬP THUỘC TÍNH	Kết quả
Username = null Client_id = null	Tất cả mọi người đều có thể truy cập vào ngữ cảnh ứng dụng.  Cặp giá trị này cũng sử dụng để cho ngữ cảnh ứng dụng dựa trên phiên truy cập (thường thì không định nghĩa, 2 thuộc tính tự null).
Username = <giá trị> Client_id = null	Cho phép một ngữ cảnh ứng dụng có thể được truy cập bởi nhiều session khác nhau, miễn là cùng một username.
Username = null Client_id = <giá trị>	Cho phép một ngữ cảnh ứng dụng được truy cập bởi nhiều session của nhiều người dùng khác nhau.
Username = <giá trị> Client_id = <giá trị>	Có 2 trường hợp:  - Lightweight users: Nếu người dùng không có một tài khoản trong CSDL, thuộc tính username trở thành chủ của các kết nối. Thuộc tính client_id trở thành liên kết với người dùng CSDL không đăng nhập.  - Database users: đối với người dùng là một người có tài khoản thì sẽ có thể sử dụng cho stateless web session.



#### D. KIỂM TRA TÁC DỤNG POLICY: VIEW V\$VPD\_POLICY

V\$VPD\_POLICY hiển thị tất cả các chính sách và các vị từ bảo mật liên quan đến các con trỏ hiện tại nằm trong bộ đệm.

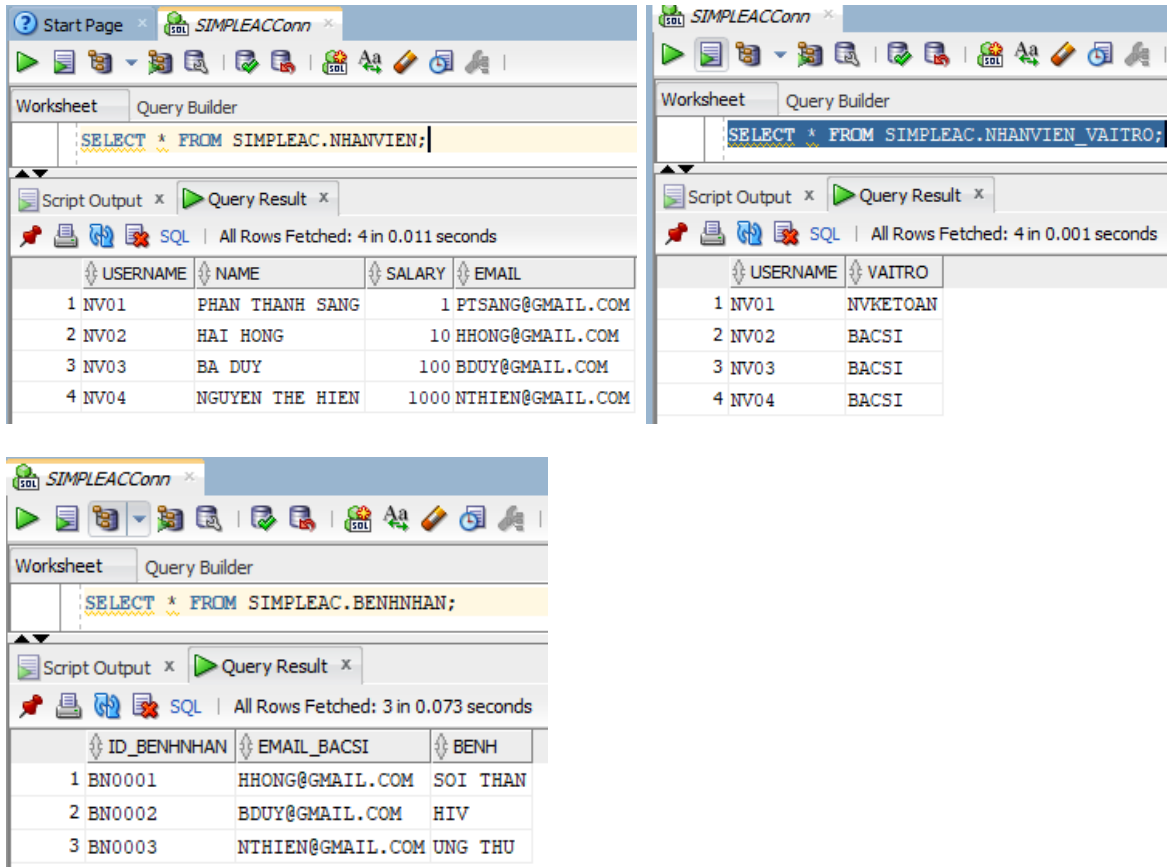
Chi tiết trong bảng sau:

Tên	Loại dữ liệu	Mô tả
<b>ADDRESS</b>	RAW (4   8)	Địa chỉ con trỏ
<b>PARADDR</b>	RAW (4   8)	Địa chỉ con trỏ tạo ra con trỏ Address
<b>SQL_HASH</b>	NUMBER	Số hash câu SQL
<b>SQL_ID</b>	VARCHAR2(13)	Id SQL
<b>CHILD_NUMBER</b>	NUMBER	Số con trỏ con mà Paraddr tạo ra
<b>OBJECT_OWNER</b>	VARCHAR2(30)	Người sở hữu đối tượng gắn với chính sách
<b>OBJECT_NAME</b>	VARCHAR2(30)	Tên của đối tượng gắn với chính sách
<b>POLICY_GROUP</b>	VARCHAR2(30)	Tên của nhóm chính sách
<b>POLICY</b>	VARCHAR2(30)	Tên của chính sách
<b>POLICY_FUNCTION_OWNER</b>	VARCHAR2(30)	Người sở hữu hàm gắn với chính sách
<b>PREDICATE</b>	VARCHAR2(4000)	Vị từ trả về của chính sách (giới hạn ở 4000 bytes)
<b>CON_ID</b>	NUMBER	0: được sử dụng cho các dòng có dữ liệu liên quan đến toàn bộ Container Database. Đồng thời cũng được sử dụng cho các dòng dữ liệu không liên quan đến Container Database. 1: Giá trị này được sử dụng cho các dòng dữ liệu có liên quan đến thư mục root. n: số dòng chứa dữ liệu liên quan đến ID container.

## IV. DEMO VPD, APPLICATION CONTEXT VÀ VPD

### A. VPD

#### 1. TÌNH HUỐNG



The first screenshot shows a query: `SELECT * FROM SIMPLEAC.NHANVIEN;` with the following results:

	USERNAME	NAME	SALARY	EMAIL
1	NV01	PHAN THANH SANG	1	PTSANG@GMAIL.COM
2	NV02	HAI HONG	10	HHONG@GMAIL.COM
3	NV03	BA DUY	100	BDUY@GMAIL.COM
4	NV04	NGUYEN THE HIEN	1000	NTHIEN@GMAIL.COM

The second screenshot shows a query: `SELECT * FROM SIMPLEAC.NHANVIEN_VAITRO;` with the following results:

	USERNAME	VAITRO
1	NV01	NVKETOAN
2	NV02	BACSI
3	NV03	BACSI
4	NV04	BACSI

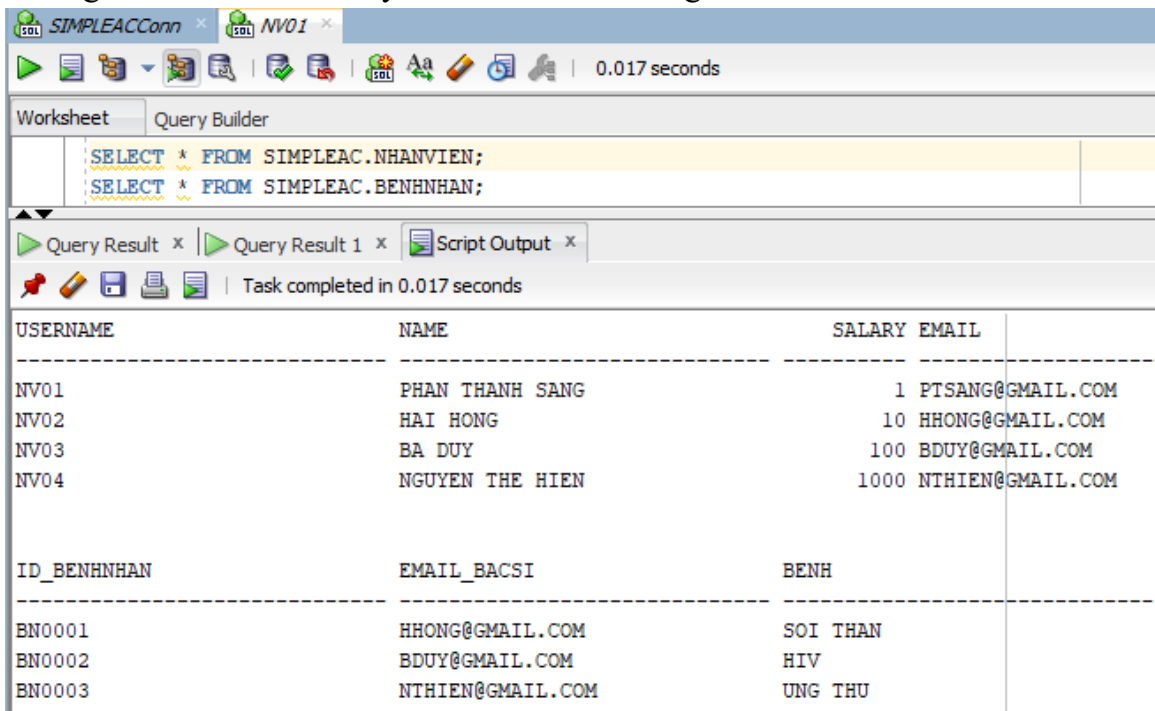
The third screenshot shows a query: `SELECT * FROM SIMPLEAC.BENHNHAN;` with the following results:

	ID_BENHNHAN	EMAIL_BACSI	BENH
1	BN0001	HHONG@GMAIL.COM	SOI THAN
2	BN0002	BDUY@GMAIL.COM	HIV
3	BN0003	NTHIEN@GMAIL.COM	UNG THU

- Một người dùng có tên đăng nhập SIMPLEAC có một cơ sở dữ liệu chứa 3 bảng NHANVIEN, NHANVIEN\_VAITRO, BENHNHAN. Trong đó có 4 tài khoản NV01, NV02, NV03, NV04.
- Ta có theo bảng NHANVIEN\_VAITRO thì NV01 là NVKETOAN (nhân viên kế toán). 3 người còn lại là BACSI (bác sĩ).
- Ta phải cấp quyền cho 4 nhân viên để họ làm việc. Như NVKETOAN phải xem hết được các dòng của bảng NHANVIEN để tính lương cho các nhân viên, BACSI phải xem được các dòng bệnh nhân mà mình phụ trách để chuẩn đoán bệnh.

```
=====
grant select on SIMPLEAC.NHANVIEN to NV01;
grant select on SIMPLEAC.NHANVIEN to NV02;
grant select on SIMPLEAC.NHANVIEN to NV03;
grant select on SIMPLEAC.NHANVIEN to NV04;
grant select on SIMPLEAC.NHANVIEN_VAITRO to NV01;
grant select on SIMPLEAC.NHANVIEN_VAITRO to NV02;
grant select on SIMPLEAC.NHANVIEN_VAITRO to NV03;
grant select on SIMPLEAC.NHANVIEN_VAITRO to NV04;
grant select on SIMPLEAC.BENHNHAN to NV01;
grant select on SIMPLEAC.BENHNHAN to NV02;
grant select on SIMPLEAC.BENHNHAN to NV03;
grant select on SIMPLEAC.BENHNHAN to NV04;
=====
```

- 
- Nhưng sau khi Grant ta thấy có vấn đề về sự riêng tư về dữ liệu.



The screenshot shows the SQL Developer interface with a query window titled 'SIMPLEACConn' and 'NV01'. The query is:

```
SELECT * FROM SIMPLEAC.NHANVIEN;
SELECT * FROM SIMPLEAC.BENHNHAN;
```

The query results are displayed in two tables. The first table shows the results of the first query, and the second table shows the results of the second query.

USERNAME	NAME	SALARY	EMAIL
NV01	PHAN THANH SANG	1	PTSANG@GMAIL.COM
NV02	HAI HONG	10	HHONG@GMAIL.COM
NV03	BA DUY	100	BDUY@GMAIL.COM
NV04	NGUYEN THE HIEN	1000	NTHIEN@GMAIL.COM

ID_BENHNHAN	EMAIL_BACSI	BENH
BN0001	HHONG@GMAIL.COM	SOI THAN
BN0002	BDUY@GMAIL.COM	HIV
BN0003	NTHIEN@GMAIL.COM	UNG THU

- 
- NV01 là kế toán nhưng lại có thể xem toàn bộ bảng BENHNHAN khi họ không có vai trò là BACSI và email không đúng với EMAIL\_BACSI phụ trách.

SIMPLEACConn x NV01 x NV02 x

0.021 seconds

Worksheet Query Builder

```
SELECT * FROM SIMPLEAC.NHANVIEN;
SELECT * FROM SIMPLEAC.BENHNHAN;
```

Script Output x

Task completed in 0.021 seconds

USERNAME	NAME	SALARY	EMAIL
NV01	PHAN THANH SANG	1	PTSANG@GMAIL.COM
NV02	HAI HONG	10	HHONG@GMAIL.COM
NV03	BA DUY	100	BDUY@GMAIL.COM
NV04	NGUYEN THE HIEN	1000	NTHIEN@GMAIL.COM

ID_BENHNHAN	EMAIL_BACSI	BENH
BN0001	HHONG@GMAIL.COM	SOI THAN
BN0002	BDUY@GMAIL.COM	HIV
BN0003	NTHIEN@GMAIL.COM	UNG THU

- NV02 là bác sĩ phụ trách bệnh nhân dòng đầu tiên bị sỏi thận nhưng lại có thể thấy luôn thông tin bệnh nhân của các bác sĩ khác và thông tin bệnh của các bệnh nhân khác? NV02 là bác sĩ còn có thể xem luôn lương của các bác sĩ khác?
- Liệu như vậy có hợp lý? Ta có nên để tình trạng như vậy không kiểm soát? Câu trả lời là không vì nó ảnh hưởng đến sự riêng tư của dữ liệu ở các trường nhạy cảm như LƯƠNG, BỆNH trong ví dụ. Ngoài ra trong thực tế còn nhiều trường nhạy cảm mà ta cần quan tâm như trường số thẻ tín dụng trong bảng nhân viên để trả lương, các triệu chứng hay bệnh nhạy cảm của các bệnh nhân, điểm của các sinh viên, ... Việc bảo vệ đăng nhập vào database là cần thiết nhưng việc kiểm soát dữ liệu mà người dùng có thể tiếp cận được cũng vô cùng cần thiết nên ta phải đề ra được chính sách giải quyết cho các trường hợp như vậy.

## 2. CÁC CHÍNH SÁCH CẦN THIẾT LẬP CHO VPD

- Để kiểm soát dữ liệu trên mức dòng ta có thể tạo View cho mỗi nhân viên và cấp View đó cho người dùng. Trong ví dụ ta có thể tạo 4 view cho các bảng đó để giới hạn dữ liệu người dùng được xem. Nhưng nếu như ta có 1000 người dùng thì việc tạo View có khả thi? À có nhưng sẽ khó kiểm soát từng view như VIEW của các nhân viên kế toán xem hết bảng còn VIEW của nhân viên BACSI chỉ xem được lương của bản thân. Nếu một nhân viên đổi VAITRO thì ta phải tìm tới View đó sau đó thay đổi câu lệnh view và làm vậy cho đến hết tất cả các bảng có sự khác biệt giữa các vai trò trong hệ thống. Ta thấy ngay dùng View kiểm soát điều này

khá khó khăn và phức tạp. Ta cần một giải pháp giải quyết vấn đề cho từng VAITRO, với người dùng này thì ta cho phép làm những điều này, với vai trò khác thì cho làm những điều khác. Lúc đó ta sẽ có thể dùng đến VPD để giải quyết.

- Trong ví dụ: ta cần 2 chính sách VPD để giải quyết
  - Thứ nhất là các NVKETOAN có thể xem lương của tất cả mọi người, các BACSI có thể xem lương của chính bản thân mình.
  - Thứ 2 là chỉ những nhân viên có vai trò là BACSI chỉ được xem thông tin của các bệnh nhân mà mình phụ trách.

### 3. CHÍNH SÁCH VPD ĐẦU TIÊN

1. Ta tạo ra hàm trả về vị từ cho chính sách tên là VPD\_NHANVIEN

```

=====
Create or replace function VPD_NHANVIEN(schema varchar2,object varchar2)
return varchar2
as
  user varchar2(100);
begin
  if ((SYS_CONTEXT('userenv', 'SESSION_USER'))='SIMPLEAC') then
    return '';
  else
    user:= SYS_CONTEXT('userenv', 'SESSION_USER');
    return 'USERNAME = user OR (''NVKETOAN'' IN (SELECT NVVT.VAITRO FROM SIMPLEAC.NHANVIEN_VAITRO NVVT
      WHERE NVVT.USERNAME = USER));'
  end if;
end;
/
=====

if ((SYS_CONTEXT('userenv', 'SESSION_USER'))='SIMPLEAC') then
  return '';
else

```

Nếu là người dùng SIMPLEAC thì trả về vị từ là '' tương đương với ko có điều kiện khi select dòng và sẽ trả về toàn bộ bảng.

```

else
  user:= SYS_CONTEXT('userenv', 'SESSION_USER');
  return 'USERNAME = user OR (''NVKETOAN'' IN (SELECT NVVT.VAITRO FROM SIMPLEAC.NHANVIEN_VAITRO NVVT
    WHERE NVVT.USERNAME = USER));'
end if;

```

Nếu khác thì trả về vị từ cột USERNAME = người dùng đang đăng nhập HOẶC 'NVKETOAN' có trong select dòng trong bảng NVVT với CỘT USERNAME = tên người dùng đang đăng nhập.



2. Ta áp dụng HÀM trả về vị từ đã viết VPD\_NHANVIEN vào trong bảng NHANVIEN của người dùng SIMPLEAC.

```

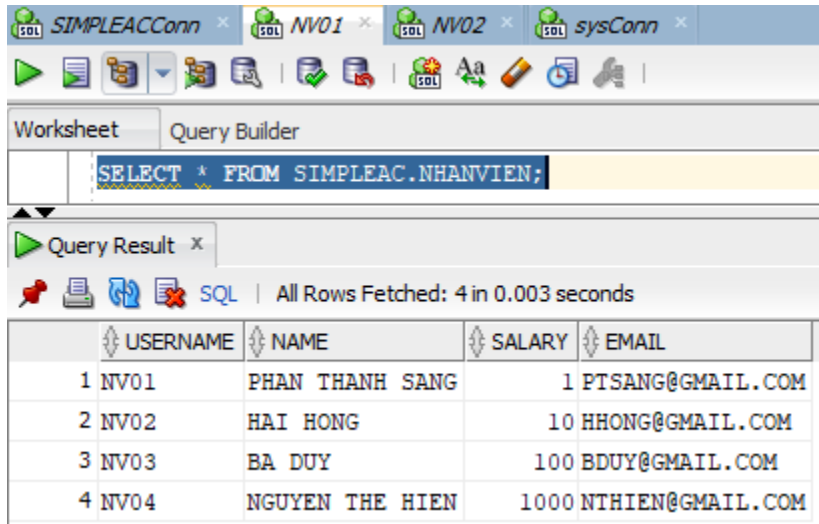
=====
BEGIN
  DBMS_RLS.ADD_POLICY
  (
    OBJECT_SCHEMA      => 'SIMPLEAC',
    OBJECT_NAME        => 'NHANVIEN',
    POLICY_NAME        => 'VPD_NHANVIEN_POLICY',
    POLICY_FUNCTION     => 'VPD_NHANVIEN',
    SEC_RELEVANT_COLS   => 'SALARY',
    SEC_RELEVANT_COLS_OPT => DBMS_RLS.ALL_ROWS
  );
END;
/
=====

```

Ta thấy Object\_schema = 'SIMPLEAC', cũng là tên người dùng đang chứa bảng NHANVIEN. Object\_name = 'NHANVIEN' là đối tượng áp dụng chính sách VPD lên. Policy\_name = 'VPD\_NHANVIEN\_POLICY' là tên của chính sách, đặt sao cho dễ nhớ để gỡ chính sách ra khi cần. Policy\_function = 'VPD\_NHANVIEN' là hàm trả về vị từ ta đã viết ở bước 1. Sec\_relevant\_cols = 'SALARY' nghĩa là khi truy vấn có tới cột SALARY trong đối tượng NHANVIEN thì sẽ kích hoạt VPD. Do bảng NHANVIEN chỉ cần bảo vệ cột SALARY nên ta có thể thêm vào nếu muốn, không có cũng được. Sec\_relevant\_cols\_opt = 'DBMS.ALL\_ROWS' có nghĩa là chính sách khi thi hành sẽ thực hiện với các cột có dữ liệu nhạy cảm sẽ xử lý làm sao? Nếu như không có dòng này thì các dòng có dữ liệu mà người dùng không được phép truy cập sẽ được ẩn đi hoàn toàn. Nhưng nếu có dòng này thì tất cả các dòng đều được load lên và chỉ cột nhạy cảm được ghi là "NULL".



### 3. Kết quả:

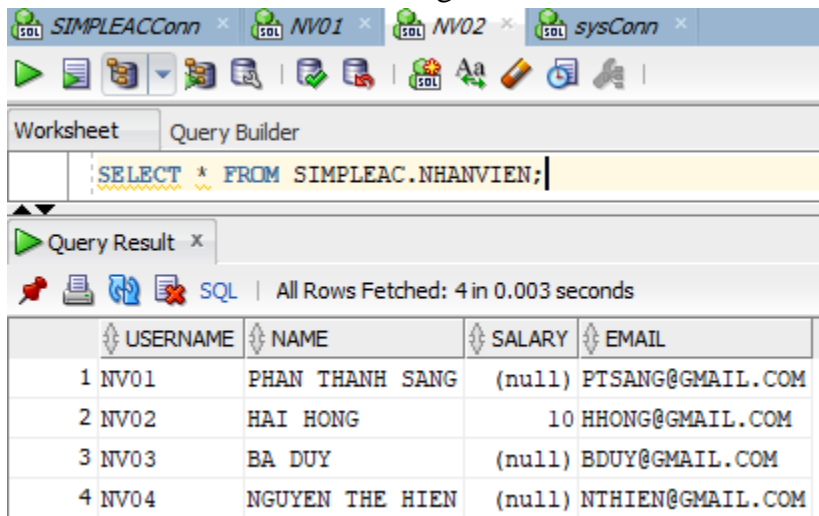


Query Result x

SQL | All Rows Fetched: 4 in 0.003 seconds

	USERNAME	NAME	SALARY	EMAIL
1	NV01	PHAN THANH SANG	1	PTSANG@GMAIL.COM
2	NV02	HAI HONG	10	HHONG@GMAIL.COM
3	NV03	BA DUY	100	BDUY@GMAIL.COM
4	NV04	NGUYEN THE HIEN	1000	NTHIEN@GMAIL.COM

- NV01 xem được tất cả các dòng NHANVIEN do có vai trò là KETOAN.



Query Result x

SQL | All Rows Fetched: 4 in 0.003 seconds

	USERNAME	NAME	SALARY	EMAIL
1	NV01	PHAN THANH SANG	(null)	PTSANG@GMAIL.COM
2	NV02	HAI HONG	10	HHONG@GMAIL.COM
3	NV03	BA DUY	(null)	BDUY@GMAIL.COM
4	NV04	NGUYEN THE HIEN	(null)	NTHIEN@GMAIL.COM

- NV02 không xem được các dòng SALARY không phải của nhân viên 2 do không có vai trò là NVKETOAN nên NV02 chỉ xem được các dòng có USERNAME = 'NV02' tức là dòng thứ 2. Các dòng còn lại sẽ che trường nhảy cảm bằng cách gán bằng "NULL".

## 4. CHÍNH SÁCH VPD THỨ HAI

1. Ta tạo ra hàm trả về vị từ cho chính sách tên là VPD\_BENHNHAN.

```

=====
Create or replace function VPD_BENHNHAN(schema varchar2,object varchar2)
return varchar2
as
    user varchar2(100);
begin
    if ((SYS_CONTEXT('userenv', 'SESSION_USER'))='SIMPLEAC') then
        return '';
    else
        user:= SYS_CONTEXT('userenv', 'SESSION_USER');
        return 'EMAIL_BACSI IN (SELECT EMAIL FROM SIMPLEAC.NHANVIEN WHERE USER = USERNAME)
                AND ('BACSI' IN (SELECT NVVT.VAITRO FROM SIMPLEAC.NHANVIEN_VAITRO NVVT
                                WHERE NVVT.USERNAME = USER));'
    end if;
end;
/
=====

if ((SYS_CONTEXT('userenv', 'SESSION_USER'))='SIMPLEAC') then
    return '';
else

```

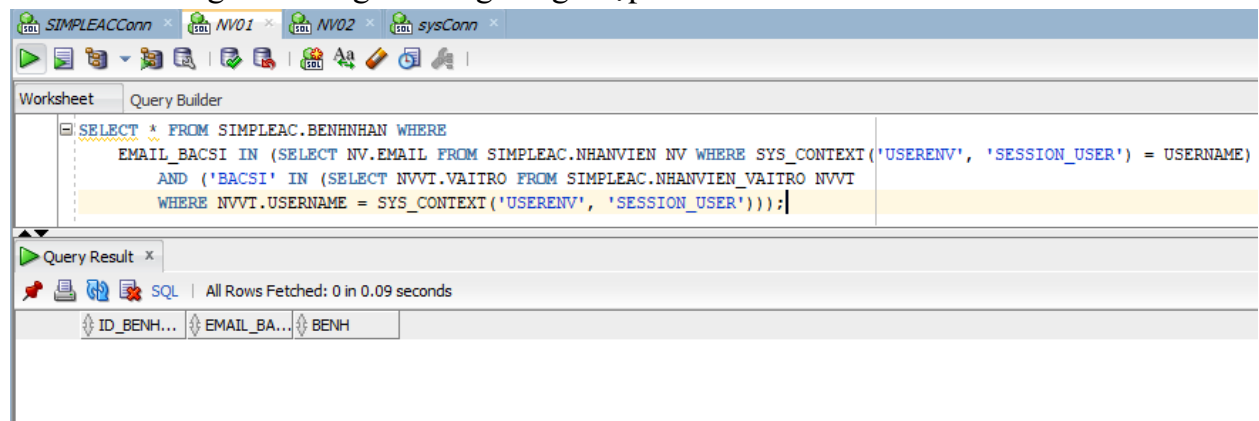
Nếu là người dung SIMPLEAC thì trả về vị từ là '' tương đương với ko có điều kiện khi select dòng và sẽ trả về toàn bộ bảng.

```

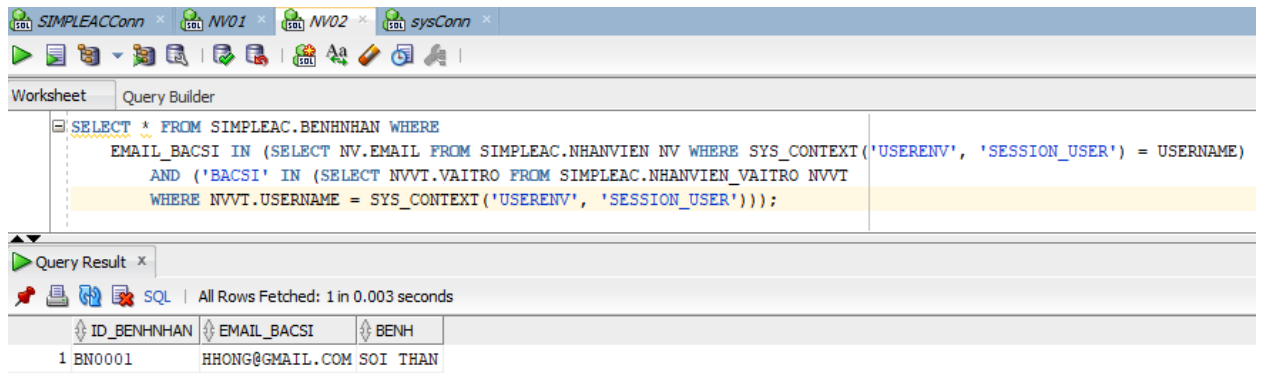
else
    user:= SYS_CONTEXT('userenv', 'SESSION_USER');
    return 'EMAIL_BACSI IN (SELECT EMAIL FROM SIMPLEAC.NHANVIEN WHERE USER = USERNAME)
            AND ('BACSI' IN (SELECT NVVT.VAITRO FROM SIMPLEAC.NHANVIEN_VAITRO NVVT
                            WHERE NVVT.USERNAME = USER));'
end if;

```

Ngắn gọn là xét một dòng bệnh nhân, nếu người đang đăng nhập có email giống như của bác sĩ quản lý bệnh nhân VÀ người đang đăng nhập có vai trò là BACSI => Trả về dòng đó cho người đang đăng nhập.



Kết quả tương tự như khi xài VPD cho user NV01.

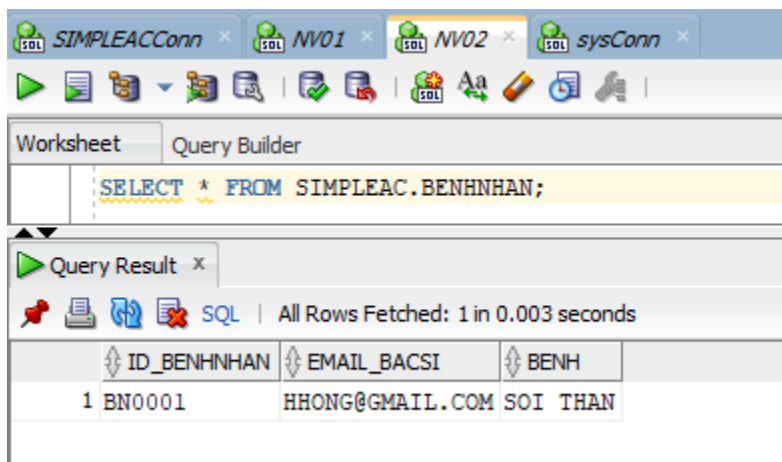
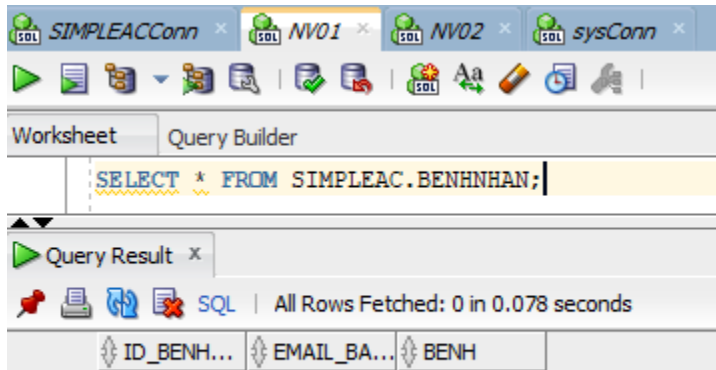


Kết quả tương tự như khi xài VPD cho user NV02.

2. Ta áp dụng HÀM trả về vị từ đã viết VPD\_BENHNHAN vào trong bảng BENHNHAN của người dùng SIMPLEAC.

```
=====
BEGIN
    DBMS_RLS.ADD_POLICY
    (
        OBJECT_SCHEMA => 'SIMPLEAC',
        OBJECT_NAME   => 'BENHNHAN',
        POLICY_NAME    => 'VPD_BENHNHAN_POLICY',
        POLICY_FUNCTION => 'VPD_BENHNHAN'
    );
END;
/
=====
```

### 3. Kết quả



Cùng một truy vấn nhưng với mỗi cá nhân cho một kết quả khác nhau.  
Giới hạn phạm vi truy cập của mỗi tài khoản dựa trên dữ liệu truy cập.

SIMPLEACConn x NV01 x NV02 x NV03 x sysConn x

0.038 seconds

Worksheet Query Builder

```
SELECT * FROM SIMPLEAC.NHANVIEN;
SELECT * FROM SIMPLEAC.BENHNHAN;
```

Script Output x

Task completed in 0.038 seconds

USERNAME	NAME	SALARY	EMAIL
NV01	PHAN THANH SANG		PTSANG@GMAIL.COM
NV02	HAI HONG		HHONG@GMAIL.COM
NV03	BA DUY	100	BDUY@GMAIL.COM
NV04	NGUYEN THE HIEN		NTHIEN@GMAIL.COM

ID_BENHNHAN	EMAIL_BACSI	BENH
BN0002	BDUY@GMAIL.COM	HIV

Thông tin được giới hạn như mong muốn qua 2 chính sách VPD.

## B. APPLICATION CONTEXT

### 1. VẤN ĐỀ

```
if ((SYS_CONTEXT('userenv', 'SESSION_USER'))='SIMPLEAC') then
    return '';
else
    user:= SYS_CONTEXT('userenv', 'SESSION_USER');
    return 'USERNAME = user OR ('NVKETOAN' IN (SELECT NVVT.VAITRO FROM SIMPLEAC.NHANVIEN_VAITRO NVVT
    WHERE NVVT.USERNAME = USER));';
end if;

else
    user:= SYS_CONTEXT('userenv', 'SESSION_USER');
    return 'EMAIL_BACSI IN (SELECT EMAIL FROM SIMPLEAC.NHANVIEN WHERE USER = USERNAME)
    AND ('BACSI' IN (SELECT NVVT.VAITRO FROM SIMPLEAC.NHANVIEN_VAITRO NVVT
    WHERE NVVT.USERNAME = USER));';
end if;
```

Hai vị từ trả về có logic đơn giản nhưng khi thiết lập lại phức tạp khó hiểu. Câu lệnh dài trên chỉ để lấy được các thông tin đơn giản như email của nhân viên, username của họ, vai trò của người đang đăng nhập với database.

```
Create or replace function VPD_PHIEUKHAM(schema varchar2,object varchar2)
return varchar2
as
user varchar2(100);
begin
if (((SYS_CONTEXT('userenv', 'SESSION_USER'))='QUANLYBENHVIEN') OR
((SYS_CONTEXT('userenv', 'SESSION_USER'))='QUANLYCHINHSAACH')) then
return '';
else
user:= SYS_CONTEXT('userenv', 'SESSION_USER');
return 'MABS = user OR ('QLCHUYENMON' IN (SELECT TKVT.VAITRO FROM QUANLYBENHVIEN.TAIKHOAN_VAITRO TKVT
WHERE TKVT.USERNAME = ' // 'user' // '))
OR ('TIEPTAN' IN (SELECT TKVT.VAITRO FROM QUANLYBENHVIEN.TAIKHOAN_VAITRO TKVT
WHERE TKVT.USERNAME = ' // 'user' // '))
OR ('QLTAINGUYENNHANSU' IN (SELECT TKVT.VAITRO FROM QUANLYBENHVIEN.TAIKHOAN_VAITRO TKVT
WHERE TKVT.USERNAME = ' // 'user' // '))
OR ('QLTAIVU' IN (SELECT TKVT.VAITRO FROM QUANLYBENHVIEN.TAIKHOAN_VAITRO TKVT
WHERE TKVT.USERNAME = ' // 'user' // '))';
end if;
end;
/
```

Mở rộng hơn, trong ví dụ ta chỉ có kiểm soát logic đơn giản nhưng khi logic trở nên phức tạp hơn, câu hỏi lặp lại nhiều lần, các thuộc tính cần thiết nằm ở nhiều bảng khác nhau và cần kết lại để tìm. Việc cài đặt VPD dựa trên các truy vấn select kết nhiều bảng làm chậm truy vấn do thực hiện lại nhiều lần nhưng đôi khi chỉ để hỏi một thuộc tính đơn giản như liệu VAITRO có là các VAITRO đặc biệt như là quản lý chính sách, quản lý phòng ban, hay trưởng phòng, thư ký. Điều đó làm giảm hiệu năng truy vấn của database.

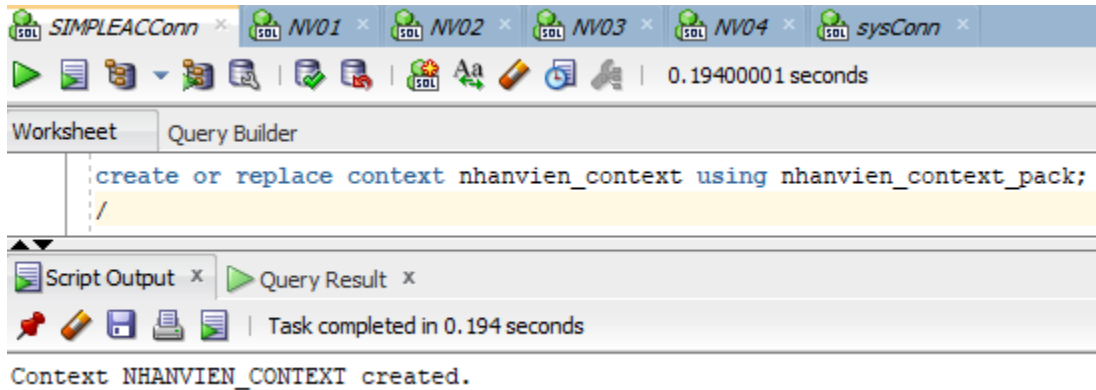
Có các nào kiểm soát các thuộc tính đó ở một nơi và khi gọi là có?

### 2. TẠO APPLICATION CONTEXT CẦN TẠO CHO DEMO

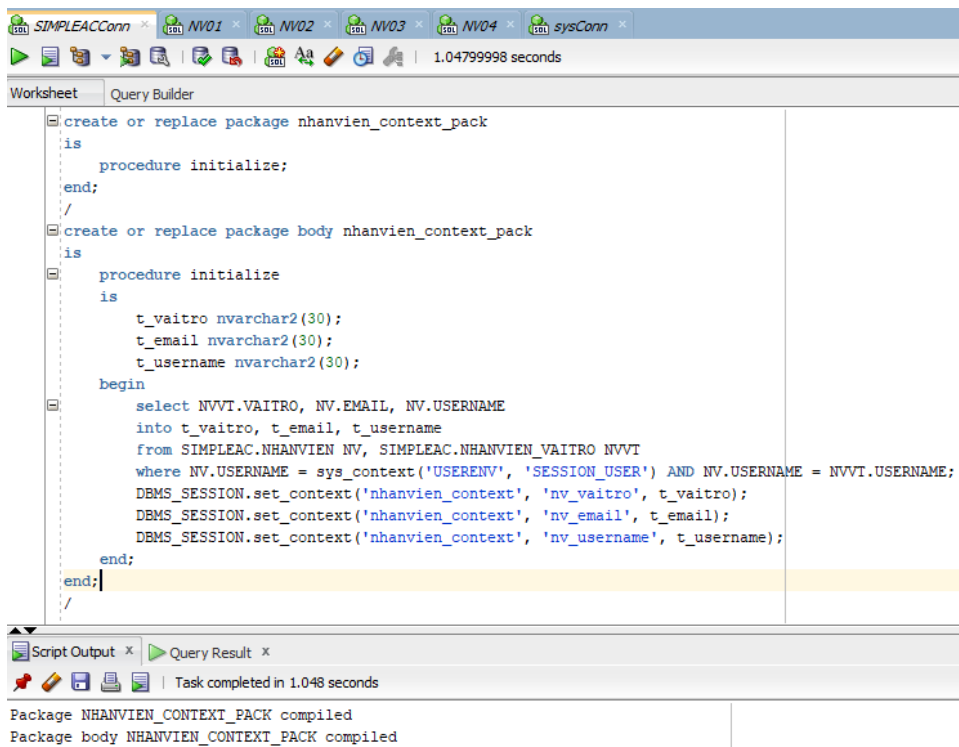


Tạo 3 thuộc tính cho một tài khoản vaitro, email, username mỗi khi đăng nhập để xác định cho các chính sách điều khiển truy cập.

Bước 1: Tạo context trong một gói



Bước 2: tạo thủ tục khởi tạo giá trị trong gói đã tạo.



Trong ví dụ là kết 2 bảng rồi cho các giá trị vai trò, email, username vào SET\_CONTEXT.

```
create or replace package body nhanvien_context_pack
is
  procedure initialize
  is
    t_vaitro nvarchar2(30);
    t_email nvarchar2(30);
    t_username nvarchar2(30);
  begin
    select NVVT.VAITRO, NV.EMAIL, NV.USERNAME
    into t_vaitro, t_email, t_username
    from SIMPLEAC.NHANVIEN NV, SIMPLEAC.NHANVIEN_VAITRO NVVT
    where NV.USERNAME = sys_context('USERENV', 'SESSION_USER') AND NV.USERNAME = NVVT.USERNAME;
    DBMS_SESSION.set_context('nhanvien_context', 'nv_vaitro', t_vaitro);
    DBMS_SESSION.set_context('nhanvien_context', 'nv_email', t_email);
    DBMS_SESSION.set_context('nhanvien_context', 'nv_username', t_username);
  end;
end;
```

	VAITRO	EMAIL	USERNAME
1	NVKETOAN	PTSANG@GMAIL.COM	NV01

Bước 3: tạo trigger chạy mỗi lần đăng nhập. Chạy thủ tục đã tạo trong packet để tạo ra 3 context cần dùng.

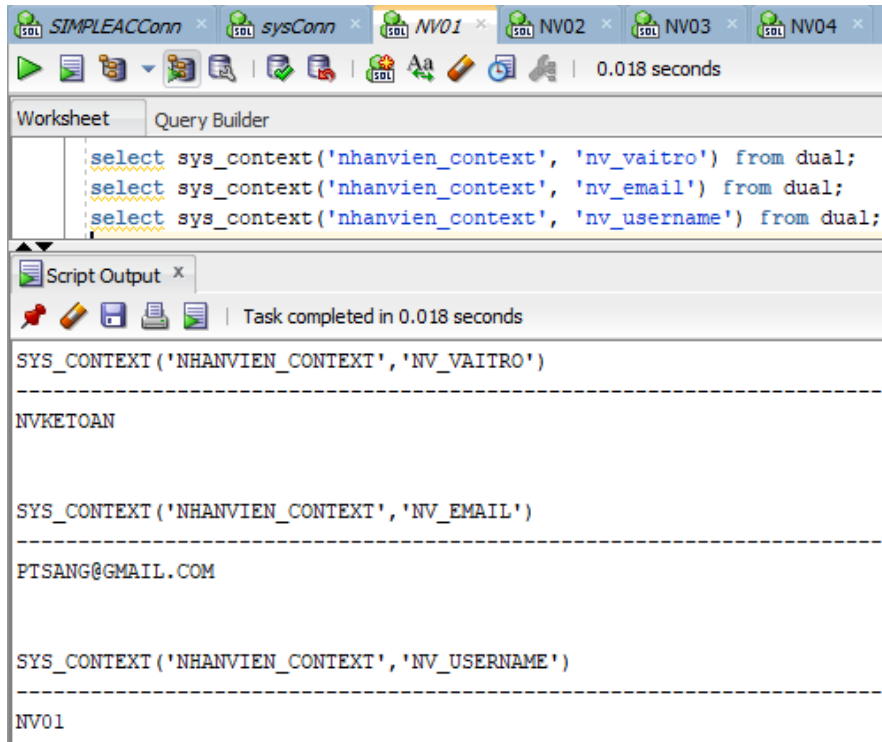
```
create or replace trigger emp_logon
after login on database
begin
  nhanvien_context_pack.initialize;
end;
```

Script Output x Query Result x

Task completed in 0.141 seconds

Trigger EMP\_LOGON compiled

## Kết quả:



Worksheet Query Builder

```
select sys_context('nhanvien_context', 'nv_vaitro') from dual;
select sys_context('nhanvien_context', 'nv_email') from dual;
select sys_context('nhanvien_context', 'nv_username') from dual;
```

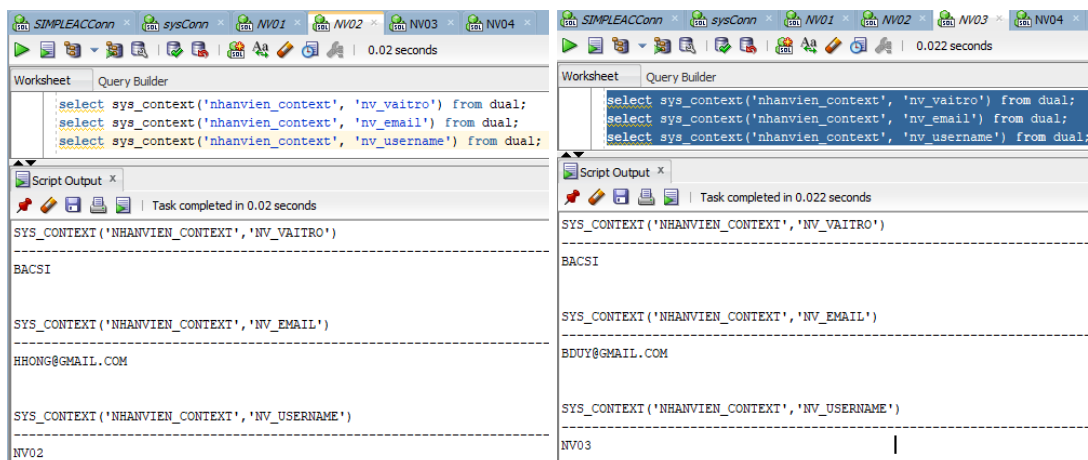
Script Output x

Task completed in 0.018 seconds

```
SYS_CONTEXT('NHANVIEN_CONTEXT','NV_VAITRO')
-----
NVKETOAN

SYS_CONTEXT('NHANVIEN_CONTEXT','NV_EMAIL')
-----
PTSANG@GMAIL.COM

SYS_CONTEXT('NHANVIEN_CONTEXT','NV_USERNAME')
-----
NV01
```



Worksheet Query Builder

```
select sys_context('nhanvien_context', 'nv_vaitro') from dual;
select sys_context('nhanvien_context', 'nv_email') from dual;
select sys_context('nhanvien_context', 'nv_username') from dual;
```

Script Output x

Task completed in 0.02 seconds

```
SYS_CONTEXT('NHANVIEN_CONTEXT','NV_VAITRO')
-----
BACSI

SYS_CONTEXT('NHANVIEN_CONTEXT','NV_EMAIL')
-----
HHONG@GMAIL.COM

SYS_CONTEXT('NHANVIEN_CONTEXT','NV_USERNAME')
-----
NV02
```

Worksheet Query Builder

```
select sys_context('nhanvien_context', 'nv_vaitro') from dual;
select sys_context('nhanvien_context', 'nv_email') from dual;
select sys_context('nhanvien_context', 'nv_username') from dual;
```

Script Output x

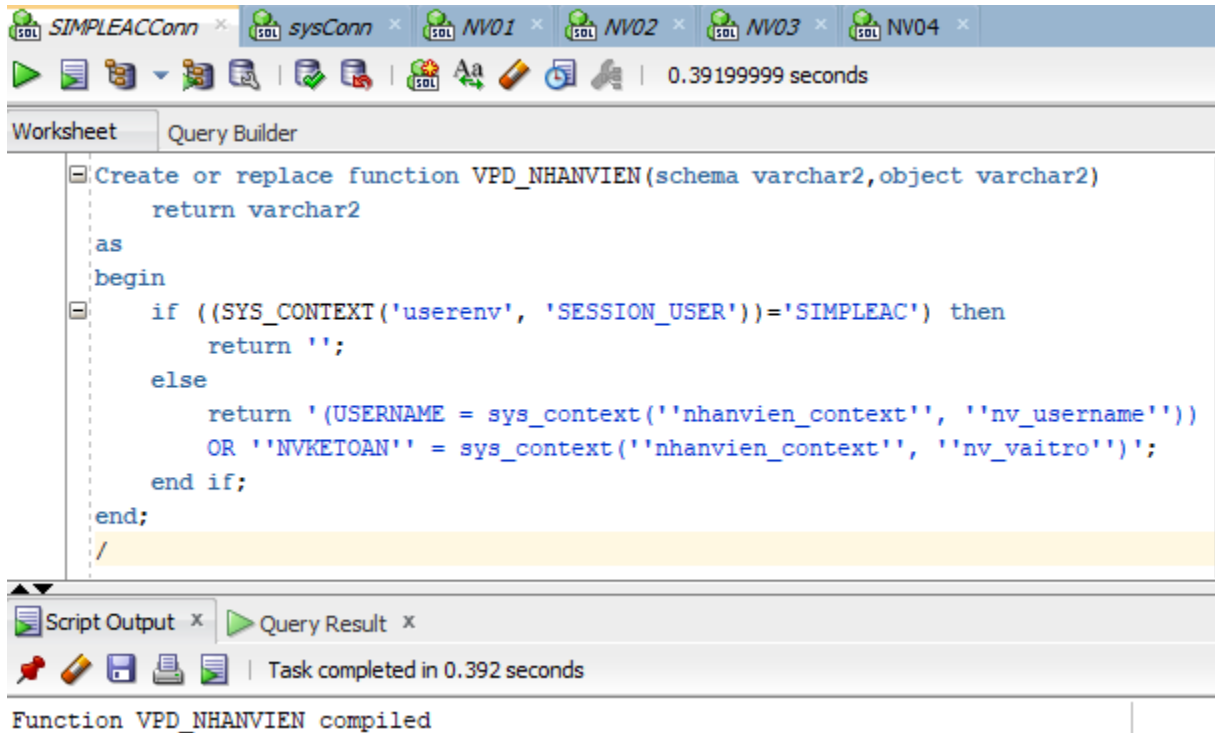
Task completed in 0.022 seconds

```
SYS_CONTEXT('NHANVIEN_CONTEXT','NV_VAITRO')
-----
BACSI

SYS_CONTEXT('NHANVIEN_CONTEXT','NV_EMAIL')
-----
BDUY@GMAIL.COM

SYS_CONTEXT('NHANVIEN_CONTEXT','NV_USERNAME')
-----
NV03
```

### 3. ỨNG DỤNG APPLICATION CONTEXT VÀO VPD

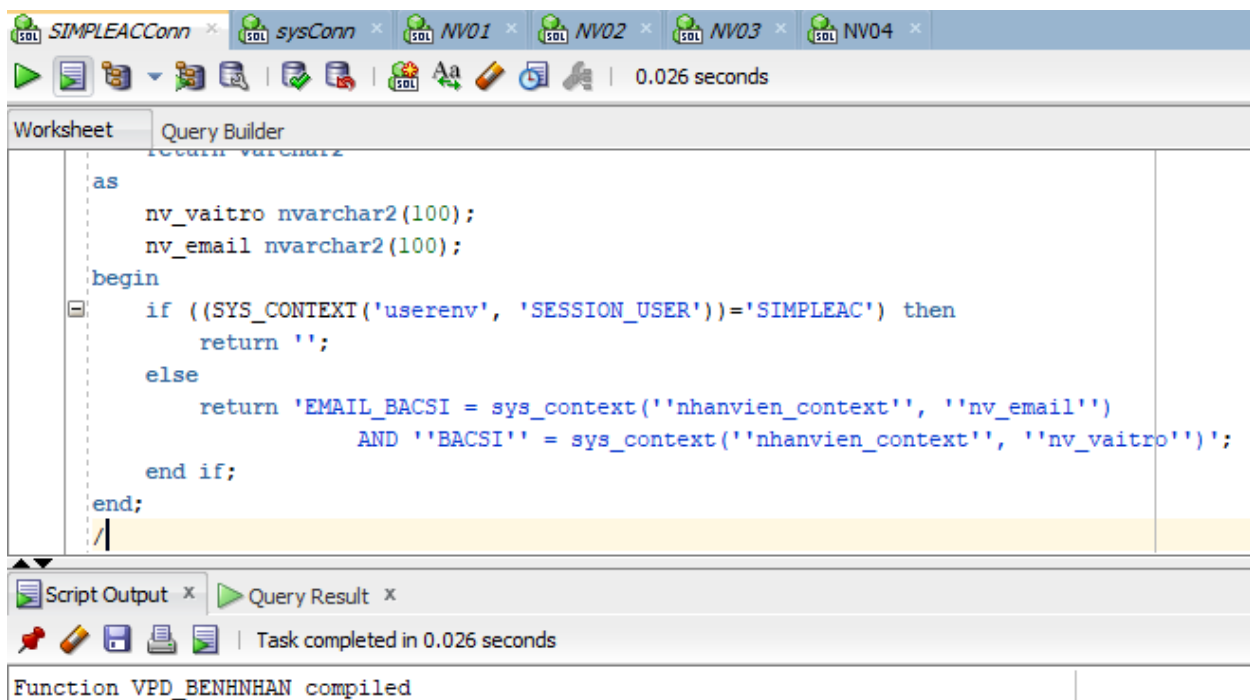


The screenshot shows the SQL Developer interface with the 'Query Builder' tab active. The main editor displays the PL/SQL code for creating or replacing the function VPD\_NHANVIEN. The code checks the session user and returns either an empty string or a VPD policy string based on the session user. The status bar at the bottom indicates 'Function VPD\_NHANVIEN compiled'.

```
Create or replace function VPD_NHANVIEN(schema varchar2,object varchar2)
return varchar2
as
begin
    if ((SYS_CONTEXT('userenv', 'SESSION_USER'))='SIMPLEAC') then
        return '';
    else
        return '(USERNAME = sys_context(''nhanvien_context'', ''nv_username''))
        OR ''NVKETOAN'' = sys_context(''nhanvien_context'', ''nv_vaitro'')';
    end if;
end;
```

Task completed in 0.392 seconds

Function VPD\_NHANVIEN compiled



The screenshot shows the SQL Developer interface with the 'Query Builder' tab active. The main editor displays the PL/SQL code for creating or replacing the function VPD\_BENHNHAN. The code checks the session user and returns either an empty string or a VPD policy string based on the session user. The status bar at the bottom indicates 'Function VPD\_BENHNHAN compiled'.

```
as
    nv_vaitro nvvarchar2(100);
    nv_email nvvarchar2(100);
begin
    if ((SYS_CONTEXT('userenv', 'SESSION_USER'))='SIMPLEAC') then
        return '';
    else
        return 'EMAIL_BACSI = sys_context(''nhanvien_context'', ''nv_email'')
        AND ''BACSI'' = sys_context(''nhanvien_context'', ''nv_vaitro'')';
    end if;
end;
```

Task completed in 0.026 seconds

Function VPD\_BENHNHAN compiled

SIMPLEACConn sysConn NV01 NV02 NV03 NV04

0.009 seconds

Worksheet Query Builder

```
SELECT * FROM SIMPLEAC.NHANVIEN;
SELECT * FROM SIMPLEAC.BENHNHAN;
```

Script Output x

Task completed in 0.009 seconds

USERNAME	NAME	SALARY	EMAIL
NV01	PHAN THANH SANG	1	PTSANG@GMAIL.COM
NV02	HAI HONG	10	HHONG@GMAIL.COM
NV03	BA DUY	100	BDUY@GMAIL.COM
NV04	NGUYEN THE HIEN	1000	NTHIEN@GMAIL.COM

no rows selected

SIMPLEACConn sysConn NV01 NV02 NV03 NV04

0.014 seconds

Worksheet Query Builder

```
SELECT * FROM SIMPLEAC.NHANVIEN;
SELECT * FROM SIMPLEAC.BENHNHAN;
```

Script Output x

Task completed in 0.014 seconds

USERNAME	NAME	SALARY	EMAIL
NV01	PHAN THANH SANG		PTSANG@GMAIL.COM
NV02	HAI HONG	10	HHONG@GMAIL.COM
NV03	BA DUY		BDUY@GMAIL.COM
NV04	NGUYEN THE HIEN		NTHIEN@GMAIL.COM

ID_BENHNHAN	EMAIL_BACSI	BENH
BN0001	HHONG@GMAIL.COM	SOI THAN

Chính sách VPD vẫn hoạt động như trước.

SIMPLEACConn sysConn NV01 NV02 NV03 NV04

sysConn

Worksheet Query Builder

```
SELECT OBJECT_OWNER, OBJECT_NAME, POLICY, PREDICATE
FROM V$VPD_POLICY;
SELECT * FROM V$VPD_POLICY;
```

Script Output x Query Result x

All Rows Fetched: 2 in 0 seconds

POLICY	PREDICATE
1 VPD_NHANVIEN_POLICY (USERNAME = sys_context('nhanvien_context', 'nv_username'))	OR 'NVKETOAN' = sys_context('nhanvien_context', 'nv_vaitro')
2 VPD_BENHNHAN_POLICY EMAIL_BACSI = sys_context('nhanvien_context', 'nv_email')	AND 'BACSI' = sys_context('nhanvien_context', 'nv_email')

- Trước khi cài: Logic cho VPD\_NHANVIEN

```
else
    user:= SYS_CONTEXT('userenv', 'SESSION_USER');
    return 'USERNAME = user      OR (''NVKETOAN'' IN (SELECT NVVT.VAITRO FROM SIMPLEAC.NHANVIEN_VAITRO NVVT
                                WHERE NVVT.USERNAME = USER));'
end if;
```

- Sau khi cài Application Context: Logic cho VPD\_NHANVIEN

```
else
    return '(USERNAME = sys_context(''nhanvien_context'', ''nv_username''))
    OR ''NVKETOAN'' = sys_context(''nhanvien_context'', ''nv_vaitro'')';
end if;
```

- Trước khi cài: Logic cho VPD\_BACSI

```
else
    user:= SYS_CONTEXT('userenv', 'SESSION_USER');
    return 'EMAIL_BACSI IN (SELECT EMAIL FROM SIMPLEAC.NHANVIEN WHERE USER = USERNAME)
    AND (''BACSI'' IN (SELECT NVVT.VAITRO FROM SIMPLEAC.NHANVIEN_VAITRO NVVT
    WHERE NVVT.USERNAME = USER));'
end if;
```

- Sau khi cài Application Context: Logic cho VPD\_BACSI

```
else
    return 'EMAIL_BACSI = sys_context(''nhanvien_context'', ''nv_email'')
    AND ''BACSI'' = sys_context(''nhanvien_context'', ''nv_vaitro'')';
end if;
```

Vị từ trả về trong chính sách VPD trở nên ngắn gọn, dễ hiểu hơn và phải thông qua một quy trình chặt chẽ mới tạo ra giúp bảo mật tốt hơn. Không còn trả về vị từ select nhiều lần, tất cả sẽ được làm khi user đăng nhập vào.



#### 4. CHI SẺ CHO TẤT CẢ NGƯỜI DÙNG APPLICATION CONTEXT GLOBAL

- Ta có tài khoản NV01 không thể select job\_role từ context global\_kda\_ctx.

```
Enter user-name: NV01/PWNV01
Last Successful login time: Thu Dec 20 2018 00:12:57 +07:00

Connected to:
Oracle Database 12c Enterprise Edition Release 12.2.0.1.0 - 64bit Production

SQL> SELECT SYS_CONTEXT('global_kda_ctx', 'job_role') job_role FROM DUAL;

JOB_ROLE
-----
```

- Đăng nhập tài khoản KDA để tạo Context globaly.

```
SQL> conn kda/123456
Connected.
SQL> create or replace context global_kda_ctx using kda_ctx_pkg ACCESSED GLOBALLY;

Context created.

SQL> CREATE OR REPLACE PACKAGE kda_ctx_pkg
2   AS
3   PROCEDURE set_kda_ctx(sec_level IN VARCHAR2);
4   PROCEDURE clear_kda_context;
5   END;
6   /

Package created.
```

- Tạo phần thân package trả về bất cứ sec\_level nào mà KDA muốn cài đặt.

```
SQL> CREATE OR REPLACE PACKAGE BODY kda_ctx_pkg
 2   AS
 3   PROCEDURE set_kda_ctx(sec_level IN VARCHAR2)
 4   AS
 5   BEGIN
 6     DBMS_SESSION.SET_CONTEXT(
 7       namespace => 'global_kda_ctx',
 8       attribute  => 'job_role',
 9       value      => sec_level);
10   END set_kda_ctx;
11   PROCEDURE clear_kda_context
12   AS
13   BEGIN
14     DBMS_SESSION.CLEAR_CONTEXT(
15       namespace      => 'global_kda_ctx',
16       attribute       => 'job_role');
17   END clear_kda_context;
18 END;
19 /
```

Package body created.

- Người dùng KDA đặt mức sec\_level ở mức NHANVIEN.

```
SQL> BEGIN
 2   kda_ctx_pkg.set_kda_ctx('NHANVIEN');
 3 END;
 4 /
```

PL/SQL procedure successfully completed.

- Những người dùng đều có thể truy cập context global\_kda\_ctx với attribute là NHANVIEN (do người dùng KDA đặt).

```
SQL> CONN NV01/PWNV01
Connected.
SQL> SELECT SYS_CONTEXT('global_kda_ctx', 'job_role') job_role FROM DUAL;

JOB_ROLE
-----
NHANVIEN

SQL> CONN NV02/PWNV02
Connected.
SQL> SELECT SYS_CONTEXT('global_kda_ctx', 'job_role') job_role FROM DUAL;

JOB_ROLE
-----
NHANVIEN
```

## V. NGUỒN THAM KHẢO

- <https://docs.oracle.com/database/121/DBSEG/vpd.htm#DBSEG245>
- [https://docs.oracle.com/cd/B28359\\_01/server.111/b28320/statviews\\_2010.htm#REFR N20164](https://docs.oracle.com/cd/B28359_01/server.111/b28320/statviews_2010.htm#REFR N20164)
- [https://docs.oracle.com/cd/B19306\\_01/network.102/b14266/policies.htm#DBSEG7000](https://docs.oracle.com/cd/B19306_01/network.102/b14266/policies.htm#DBSEG7000)
- [https://www.techonthenet.com/oracle/functions/sys\\_context.php](https://www.techonthenet.com/oracle/functions/sys_context.php)
- <http://antoanthongtin.vn/Detail.aspx?NewsID=4acca727-4e05-4c58-b3b3-c29bc361968c&CatID=8ab90f49-a562-4157-a607-d2474bf129a9>
- <https://text.123doc.org/document/2605348-tim-hieu-ve-co-so-du-lieu-rieng-ao-trong-oracle.htm>
- [https://docs.oracle.com/cd/B28359\\_01/network.111/b28531/app\\_context.htm#DBSEG011](https://docs.oracle.com/cd/B28359_01/network.111/b28531/app_context.htm#DBSEG011)
- [https://docs.oracle.com/cd/B19306\\_01/network.102/b14266/apdvctx.htm#DBSEG14000](https://docs.oracle.com/cd/B19306_01/network.102/b14266/apdvctx.htm#DBSEG14000)
- [http://www.dba-oracle.com/t\\_adv\\_plsql\\_vpd\\_application\\_contexts.htm](http://www.dba-oracle.com/t_adv_plsql_vpd_application_contexts.htm)
- [https://docs.oracle.com/cd/B19306\\_01/network.102/b14266/apdvctx.htm#i1009020](https://docs.oracle.com/cd/B19306_01/network.102/b14266/apdvctx.htm#i1009020)