Database Programming with PL/SQL

Review of Object Privileges

Objectives

This lesson covers the following objectives:

- List and explain several object privileges
- Explain the function of the EXECUTE object privilege
- Write SQL statements to grant and revoke object privileges

Purpose

You already know that one of the benefits of PL/SQL subprograms is that they can be reused in many applications.

Users can call and execute subprograms only if they have the privileges to do so.

This lesson first reviews object privileges in general, then focuses in more detail on the privileges needed to execute a PL/SQL subprogram.

What Is an Object Privilege?

An object privilege allows the use of a specific database object, such as a table, a view, or a PL/SQL procedure, by one or more database users.

When a database object is first created, only its owner (creator) and the Database Administrator are privileged to use it.

Privileges for all other users must be specifically granted (and maybe later revoked). This can be done by the object's owner or by the DBA.

What Object Privileges Are Available?

Each object has a particular set of grantable privileges. The following table lists the privileges for various objects.

Object Privilege	Table	View	Sequence	Procedure
ALTER	X		X	
DELETE	X	X		
EXECUTE				X
INDEX	X			
INSERT	X	X		
REFERENCES	X	X		
SELECT	X	X	X	
UPDATE	X	X		

What Object Privileges Are Available? (cont.)

`SELECT`, `INSERT`, `UPDATE`, and `DELETE` privileges allow the holder (the grantee) of the privilege to use the corresponding SQL statement on the object.

For example, `INSERT` privilege on the EMPLOYEES table allows the holder to `INSERT` rows into the table, but not to `UPDATE` or `DELETE` rows.

What Object Privileges Are Available? (cont.)

The `ALTER` privilege allows the grantee to `ALTER` the table, while `INDEX` privilege allows the grantee to create indexes on the table. Of course, you can automatically do this on your own tables.

The `REFERENCES` privilege allows the grantee to check for the existence of rows in a table or view using foreign key constraints.

Granting Object Privileges

Syntax:

```
GRANT object_priv [(columns)]
ON object
TO {user|role|PUBLIC}
[WITH GRANT OPTION];
```

Examples:

```
GRANT INSERT, UPDATE
ON employees TO TOM, SUSAN;

GRANT SELECT
ON departments TO PUBLIC;
```

Syntax	Definition
object_priv	Is an object privilege to be granted.
columns	Specifies a column from a table or view on which privileges are granted.
ON object	Is the object on which the privileges are granted.
user|role	Identifies the user or role to whom the privilege is granted.
PUBLIC	Grants object privileges to all users.
WITH GRANT OPTION	Allows the grantee to grant the object privileges to other users and roles.

Revoking Object Privileges

Syntax:

```
REVOKE object_priv [(columns)]
ON object
FROM {user|role|PUBLIC};
```

Examples:

```
REVOKE INSERT, UPDATE ON employees FROM TOM, SUSAN;

REVOKE SELECT ON departments FROM PUBLIC;
```

Using the EXECUTE Privilege With Stored Subprograms Example

To invoke and execute a PL/SQL subprogram, the user must be granted EXECUTE privilege on the subprogram.

```
CREATE OR REPLACE PROCEDURE add_dept ... ;
CREATE OR REPLACE FUNCTION  get_sal ...;

GRANT EXECUTE ON add_dept TO TOM, SUSAN;
GRANT EXECUTE ON get_sal  TO PUBLIC;
...
REVOKE EXECUTE ON get_sal FROM PUBLIC;
```

Referencing Objects in Subprograms

What about the objects referenced inside the subprogram? To invoke a subprogram, a user needs only EXECUTE privilege on the subprogram. He/she does NOT need any privileges on the objects referenced by SQL statements within the subprogram.

```
CREATE OR REPLACE PROCEDURE add_dept ...
IS BEGIN
...
  INSERT INTO DEPARTMENTS ... ;
...
END;

GRANT EXECUTE ON add_dept TO SUSAN;
```

The user (SUSAN) does not need INSERT (or any other privilege) on the DEPARTMENTS table.

Privileges on Referenced Objects

Someone must have privileges on the referenced objects. Who is it? The subprogram owner (creator) must hold the appropriate privileges on the objects referenced by the subprogram. The owner's privileges are checked when the subprogram is created or replaced, and also every time the subprogram is invoked. In this example, TOM creates a procedure that SUSAN needs to invoke:

```
(Table owner or DBA): GRANT INSERT ON departments TO TOM;
(Tom) CREATE OR REPLACE PROCEDURE add_dept ...
        IS BEGIN
           ...
           INSERT INTO DEPARTMENTS ... ;
           ...
        END;
(Tom) GRANT EXECUTE ON add_dept TO SUSAN;
```

Privileges on Referenced Objects (cont.)

Below is another example. BILL owns the STUDENTS and GRADES tables. HANNAH needs to create a procedure that JIEP needs to invoke:

```
(Hannah)CREATE OR REPLACE PROCEDURE student_proc ...
        IS BEGIN
           SELECT ... FROM bill.students JOIN bill.grades ...;
           UPDATE bill.students ...;
           ...
        END;

(Jiep)  BEGIN    hannah.student_proc(…);  END;
```

Who needs which privileges on which objects?

System Privileges

On the previous slide, HANNAH created a procedure. What privilege(s) did HANNAH need in order to do this?

Yes, HANNAH needs suitable object privileges on BILL's tables. She also needs the CREATE PROCEDURE system privilege:

```
(DBA) GRANT CREATE PROCEDURE TO hannah;
```

Although the name of the privilege is CREATE PROCEDURE, it also allows HANNAH to create functions and packages.

Terminology

Key terms used in this lesson included:

- `ALTER` privilege
- `EXECUTE` privilege
- `INDEX` privilege
- Object privilege
- `REFERENCES` privilege

Summary

In this lesson, you should have learned how to:

- List and explain several object privileges
- Explain the function of the EXECUTE object privilege
- Write SQL statements to grant and revoke object privileges