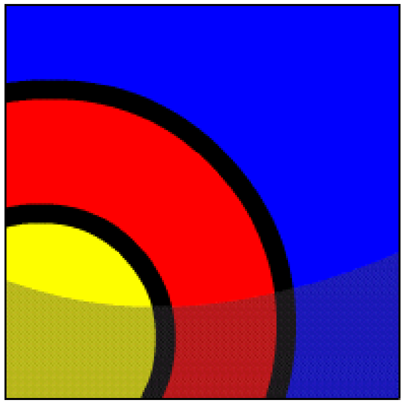
ORACLE Academy

Using Invoker’s Rights

What Will I Learn?

In this lesson, you will learn to:

- Contrast invoker's rights with definer's rights
- Create a procedure that uses invoker's rights



Why Learn It?

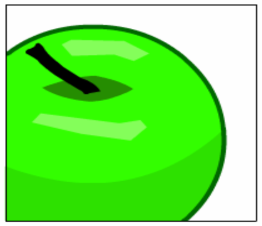
In the previous lesson, you learned that the creator (owner) of a PL/SQL subprogram must be granted the relevant object privileges on the tables that are referenced by SQL statements within the subprogram. These privileges are checked every time the program is invoked.

This way of checking privileges is called Definer's Rights.

What if you want the invoking user's privileges to be checked instead? You must understand and use Invoker's Rights.

Tell Me / Show Me

Definer's Rights Compared to Invoker's Rights



"Definer" means the owner (usually the creator) of the PL/SQL subprogram. "Invoker" means the user who calls (invokes) the subprogram.

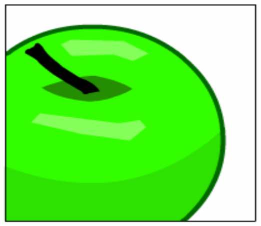
When Definer's Rights are used (the default):

- The Definer needs privileges on the database objects referenced within the subprogram.
- The Invoker (= User) only needs `EXECUTE` privilege on the subprogram.

The subprogram executes in the User's database session, but with the privileges of the Definer.

Tell Me / Show Me

Definer's Rights Compared to Invoker's Rights (contir



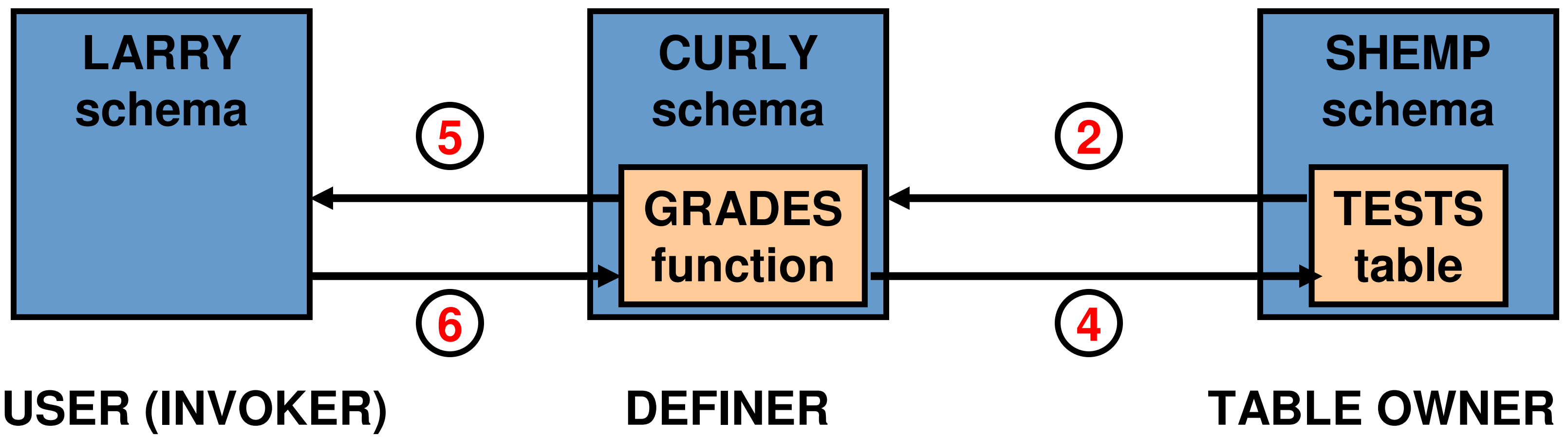
When Invoker's Rights are used:

- The Invoker needs privileges on the database objects referenced within the subprogram, as well as `EXECUTE` privilege on the procedure
- The Definer does not need any privileges.

The subprogram executes in the User's database session, with the User's privileges.

Tell Me / Show Me

An Example of Definer's Rights:



1. Shemp> `CREATE TABLE tests (score NUMBER, etc…`
2. Shemp> `GRANT SELECT ON tests TO curly;`
3. Shemp> `REVOKE SELECT ON tests FROM larry;`
4. Curly> `CREATE FUNCTION grades RETURN etc…`
   `(this function queries the SHEMP.TESTS table)`
5. Curly> `GRANT EXECUTE ON grades TO larry;`
6. Larry> `... BEGIN  v_result := curly.grades;  END;`

Tell Me / Show Me

Using Invoker's Rights

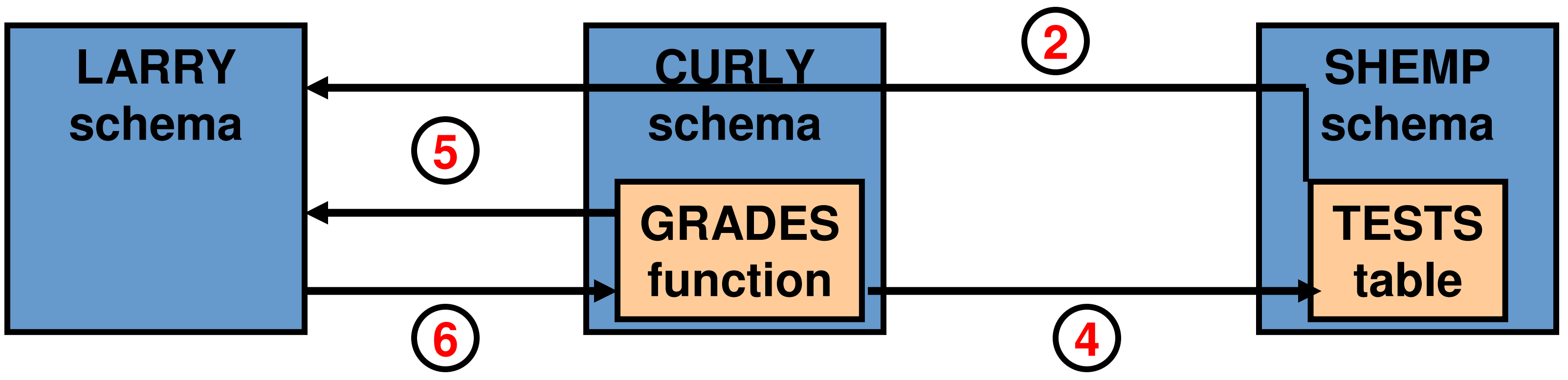
Set `AUTHID` to `CURRENT_USER`, immediately before `IS | AS`:

```
CREATE OR REPLACE FUNCTION grades
   (p_name IN VARCHAR2) RETURN NUMBER
    AUTHID CURRENT_USER IS
  v_score  NUMBER;
BEGIN
  SELECT score INTO v_score FROM shemp.tests
    WHERE key=p_name;
  RETURN v_score;
END;
```

Now who needs which privileges?

Tell Me / Show Me

Using Invoker's Rights (continued)



USER (INVOKER) DEFINER TABLE OWNER

1. Shemp> `CREATE TABLE tests (score NUMBER, etc…`
2. Shemp> `GRANT SELECT ON tests TO larry;`
3. Shemp> `REVOKE SELECT ON tests FROM curly;`
4. Curly>
```
CREATE FUNCTION grades
   RETURN ... AUTHID CURRENT_USER ...
   (this function queries the SHEMP.TESTS table)
```
5. Curly> `GRANT EXECUTE ON grades TO larry;`
6. Larry> `... BEGIN  v_result := curly.grades;  END;`

Tell Me / Show Me

Another example

```
Tom> CREATE OR REPLACE PROCEDURE add_dept ...
     AUTHID CURRENT_USER IS
     BEGIN
       INSERT INTO departments ...;
     END;
Tom> GRANT EXECUTE ON add_dept TO susan;

Susan> BEGIN   tom.add_dept(...); END;
```

Who needs `INSERT` privilege on `DEPARTMENTS`:

- If Definer's rights are used?

- If Invoker's rights are used?

Tell Me / Show Me

Which Schema Is the table In?

Look at this example:

```
Tom> CREATE TABLE tests ... ;
Tom> CREATE OR REPLACE PROCEDURE grades ... IS
       BEGIN
       ... SELECT ... FROM tests ... ;
       END;

Susan> BEGIN ... tom.grades(...); END;
```

When using Definer's rights, table-name `TESTS` is resolved in the definer's schema. This means that table `TESTS` is assumed to be in Tom's schema, so you do not need to prefix the table-name with the schema-name. This procedure compiles and executes successfully provided the relevant privileges have been granted.

Tell Me / Show Me

Now you change the procedure to use Invoker’s rights.

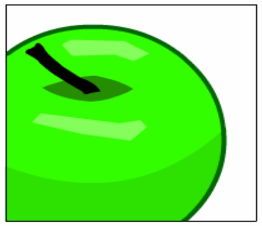
```
Tom> CREATE TABLE tests ... ;
Tom> CREATE OR REPLACE PROCEDURE grades ...
     AUTHID CURRENT_USER IS
       BEGIN
       ... SELECT ... FROM tests ... ;
       END;

Susan> BEGIN ... tom.grades(...); END;
```

Using Invoker’s rights, object-names are resolved in the Invoker’s schema. This means that `TESTS` is now assumed to be in Susan’s schema. How would you correct the procedure code to enable the procedure to work correctly? (Remember that `TESTS` is in Tom’s schema.)

Tell Me / Show Me

Terminology

Key terms used in this lesson include:



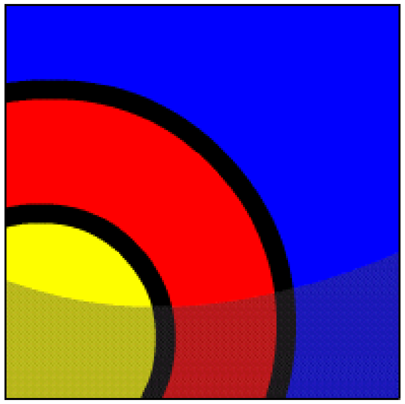
Definer's rights

Invoker's rights

Summary

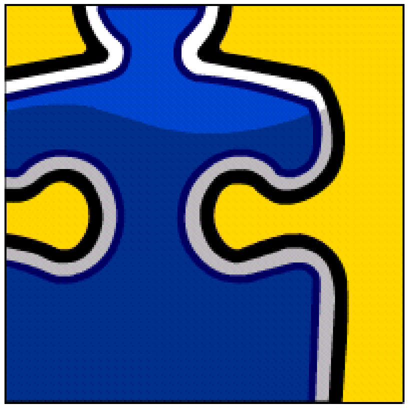
In this lesson, you learned to:

- Contrast invoker's rights with definer's rights
- Create a procedure that uses invoker's rights



Try It / Solve It

The exercises in this lesson cover the following topics:



- Contrasting invoker’s rights with definer’s rights
- Creating a procedure that uses invoker’s rights