

Invoker rights and Autonomous transactions

ORACLE

Copyright © Oracle Corporation, 2004. All rights reserved.

Objectives

After completing this lesson, you should be able to do the following:

- **Understand and use advanced function definitions**
 - Describe the invoker rights model
 - Define autonomous transactions

More on Function Definitions

- DETERMINISTIC
- PARALLEL_ENABLE
- AUTHID
- AUTONOMOUS_TRANSACTION

```
[CREATE [OR REPLACE]]FUNCTION function_name
[(parameter[, parameter]...)]
RETURN datatype}
  [ AUTHID {DEFINER | CURRENT_USER}]
  [ DETERMINISTIC] {IS | AS}
  [ PRAGMA AUTONOMOUS_TRANSACTION ;]
  [ local declarations]
BEGIN
  ...
```

The DETERMINISTIC Hint

- Can be used in functions as an optimization hint
- Indicates that the results of a function are always the same for a given set of input argument values
- Means that SQL may choose to use a cached copy of the return result rather than actually invoking the function
- Is mandatory for functions used in function-based indexes and materialized views with **ENABLE QUERY REWRITE**

DETERMINISTIC

- When returning the same result for the same input, use the DETERMINISTIC pragma.

```
CREATE OR REPLACE FUNCTION tax (amt IN NUMBER)
  RETURN NUMBER
  DETERMINISTIC
IS
BEGIN
  RETURN (amt *.08);
END tax;
```

- Do not use DETERMINISTIC for functions that produce varying results.

```
CREATE OR REPLACE FUNCTION calcAge (dateOfBirth IN DATE)
  RETURN NUMBER
IS
BEGIN
  RETURN sysdate - dateOfBirth;
END calcAge;
```

Definer Rights Versus Invoker Rights

Definer rights:

- Programs execute with the privileges of the creating user.
- A user does not require privileges on underlying objects the procedure accesses; only requires privilege to execute a procedure.
- Roles are not active.

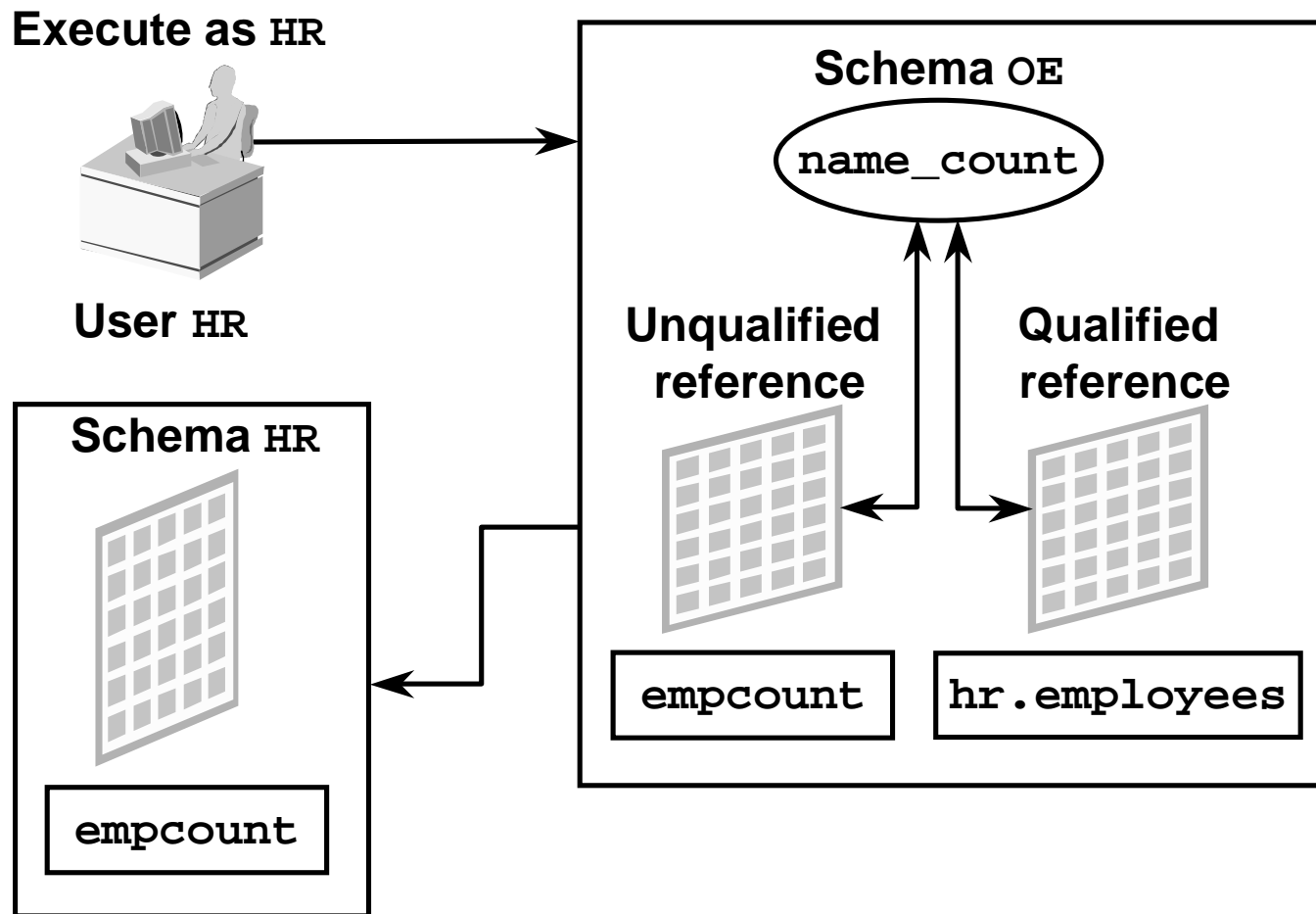
Invoker rights:

- Programs execute with the privileges of the calling user.
- A user requires privileges on the underlying objects the procedure accesses.
- Invoker's roles are active.

Invoker Rights

- **Set AUTHID to CURRENT_USER in:**
 - CREATE FUNCTION
 - CREATE PROCEDURE
 - CREATE PACKAGE
- **Names used in queries, DML, dynamic SQL, and dbms_sql are resolved in the invoker's schema.**
- **Calls to packages, functions, and procedures are resolved in the definer's schema.**

Invoker Rights



ORACLE

Copyright © Oracle Corporation, 2004. All rights reserved.

Invoker Rights

User OE:

```
EXECUTE name_count  
PL/SQL procedure successfully completed.
```

User HR:

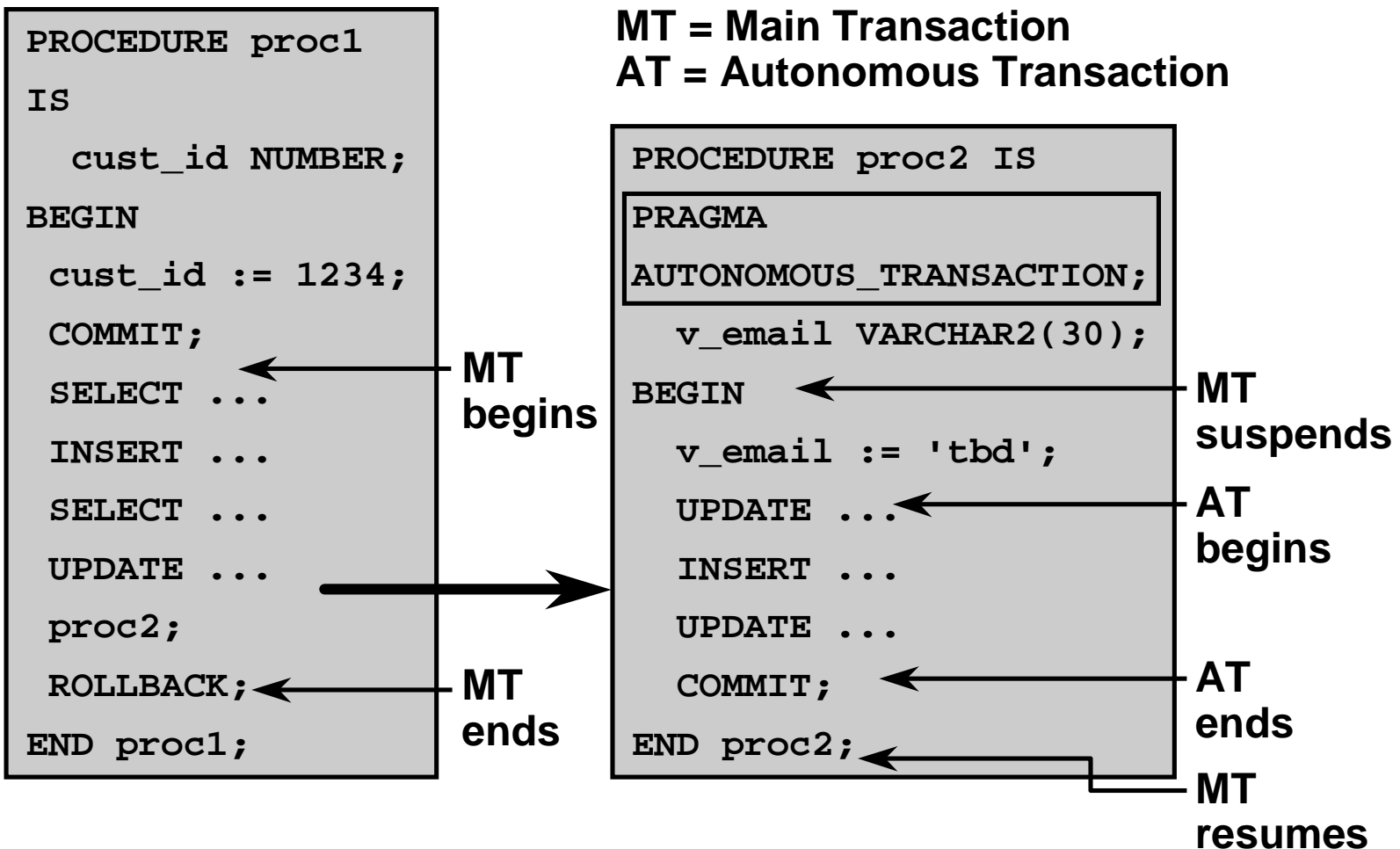
```
EXECUTE oe.name_count  
  
*  
ERROR at line 1:  
ORA-00942: table or view does not exist  
ORA-06512: at "OE.NAME_COUNT", line 11  
ORA-06512: at line 1
```

- **EMPCOUNT table does not exist in the HR schema.**

ORACLE

Copyright © Oracle Corporation, 2004. All rights reserved.

Autonomous Transactions



ORACLE

Autonomous Transactions

- **Automatic transaction is an independent transaction started by another transaction.**
- **Automatic transaction is independent of the main transaction; not nested transactions.**
- **Automatic transaction does not roll back if the main transaction rolls back.**
- **In automatic transactions changes become visible to other transactions upon a commit.**
- **Once completed, the calling transaction is visible again.**
- **Only individual routines can be marked autonomous.**
- **You cannot mark a nested PL/SQL block as autonomous.**

Autonomous Transactions

```
CREATE OR REPLACE PROCEDURE upd_cust_credit
(p_cust_id NUMBER)
IS
    PRAGMA AUTONOMOUS_TRANSACTION;
    v_email VARCHAR2(30);
BEGIN
    SELECT  cust_email
      INTO  v_email
    FROM    customers
    WHERE   customer_id = p_cust_id;
    UPDATE customers
      SET   credit_limit = credit_limit * 1.1
    WHERE  customer_id = p_cust_id;
    -- could have code to send an email to the customer
    -- to notify them of the credit limit change ...
    COMMIT;
END;
/;
```

ORACLE

Copyright © Oracle Corporation, 2004. All rights reserved.

Autonomous Transactions

Example

```
CREATE OR REPLACE PROCEDURE add_order_entries
(p_ord_id NUMBER, p_prod_id NUMBER,
 p_quantity NUMBER, p_cust_id NUMBER)
IS
  v_line_no      NUMBER(5);
  v_total        NUMBER; ...
BEGIN
  SELECT  MAX(NVL(line_item_id, 0) + 1) ...
  SELECT  list_price ...
  INSERT INTO order_items ...
  SELECT SUM(unit_price * NVL(quantity, 1)) ...
  UPDATE orders ...
  upd_cust_credit (p_cust_id);
  COMMIT;
END;
/
```

← Calls the Autonomous Transaction

ORACLE

Copyright © Oracle Corporation, 2004. All rights reserved.

Summary

In this lesson, you should have learned how to:

- **Use the DETERMINISTIC and PARALLEL_ENABLE purity features, invoker rights model and autonomous transactions**

Practice Overview

This practice covers the following topics:

- **Implement design considerations**
- **Define local subprograms**
- **Use the invoker rights model**
- **Create an autonomous transaction**