DUMPSARENA

Oracle Database 12c: Advanced PL/SQL

Oracle 1z0-148

Version Demo

Total Demo Questions: 10

Total Premium Questions: 106

Buy Premium PDF

https://dumpsarena.co

sales@dumpsarena.co

sales@dumpsarena.co
dumpsarena.co

QUESTION NO: 1

In which two situations will a result-cached function execute even though no change was made on a dependent table or view? (Choose two.)

A. when a function was created with invoker's rights using AUTHID CURRENT_USER

B. when the database administrator has disabled the use of the result cache

C. when the results for the function have aged out of the cache

D. when the RELIES_ON clause was not used while creating the function

E. when a new session was opened to invoke the function which is already cached

ANSWER: B C

QUESTION NO: 2

Examine this block of code used to calculate the price increase for all the productivity by 1% and then by 2%.

```
DECLARE
    incr_percent NUMBER := .01;
    CURSOR pdt_cur IS
        SELECT prod_name, (prod_min_price* incr_percent) FROM pdts;
BEGIN
    FOR pdt_rec IN pdt_cur
    LOOP
        DBMS_OUTPUT.PUT_LINE ('PROD NAME' || pdt_rec.prod_name || 'PRICE
                    INCREASE AMT' || pdt_rec.(prod_min_price * incr_percent) );
         incr_percent := incr_percent + .01;
     END LOOP;
END;
/
```

What will be the outcome on execution?

A. It will give an error because the calculated column in the cursor is not using a column alias in this block.

B. It will go into an endless loop because the loop exist condition is missing.

C. It will display the price increase by 1% only for all the products.

D. It will display the price increase by 1% only for the first product.

E. It will give an error because PDT_REC is not declared.

ANSWER: A

QUESTION NO: 3

Examine this code:

```
CREATE FUNCTION emp_policy_fn (v_schema IN VARCHAR2, v_objname IN VARCHAR2)
RETURN VARCHAR2 AS
     con VARCHAR2 (200);
BEGIN
    con:= 'deptno= 30';
    RETURN con;
END emp_policy_fn;
/
BEGIN
     DBMS_RLS.ADD_POLICY (
       object_schema =>'schott',
       object_name=> 'emp',
       policy_name=> 'emp_policy',
       policy_function=>'emp_policy_fn',
       update_check=> TRUE,
       statement_types => 'SELECT, UPDATE',
       sec_relevant_cols=> 'sal, comm');
END;
/
```

Examine this DML statement executed in the SCOTT schema:

UPDATE emp SET comm = 1000 WHERE deptno= 20;

What is the outcome after executing this statement?

A. COMM is set to 1000 for all records in the EMP table where DEPTNO = 30.

B. The statement executes successfully but no rows are updated.

C. COMM is set to 1000 for all records in the EMP table where DEPTNO=20.

D. The statement fails with error ORA-28115: policy with check option violation.

ANSWER: D

QUESTION NO: 4

Examine this function call:

cur_num := DBMS_SQL.TO_CURSOR_NUMBER (cur_val);

Which two statements are true? (Choose two.)

A. CUR_VAL must be opened after this line is executed in the PL/SQL block.

B. CUR_VAL must be a weakly typed cursor variable.

C. CUR_VAL can be either a strongly or weakly typed cursor variable.

D. CUR_VAL must be opened before this line is executed in the PL/SQL block.

E. The PL/SQL block can continue to use the cursor variable after this line is executed.

ANSWER: C D

Explanation:

Reference: https://docs.oracle.com/cd/B28359_01/appdev.111/b28370/dynamic.htm#LNPLS011

QUESTION NO: 5

You are connected as SCOTT who has an EMP table with this structure:

```
Name                 Null?    Type
-------------------- -------- --------------------
EMPNO                NOT NULL NUMBER(4)
ENAME                         VARCHAR2(10)
JOB                           VARCHAR2(9)
MGR                           NUMBER(4)
HIREDATE                      DATE
SAL                           NUMBER(7,2)
COMM                          NUMBER(7,2)
DEPTNO                        NUMBER(2)
```

Examine this code:

```
  CREATE FUNCTION emp_policy_fn (v_schema IN VARCHAR2, v_objname IN VARCHAR2) RETURN
VARCHAR2 AS
     con VARCHAR2(200);
  BEGIN
     con := 'deptno = 30';
     RETURN con;
  END emp_policy_fn;
  /
  BEGIN
     DBMS_RLS.ADD_POLICY (
        object_schema => 'scott',
        object_name => 'emp',
        policy_name => 'emp_policy',
        policy_function => 'emp_policy_fn',
        sec_relevant_cols => 'sal, comm' );
  END;
  /
  CREATE EDITION a1;
  ALTER SESSION SET EDITION = a1;
  CREATE OR REPLACE FUNCTION emp_policy_fn (v_schema IN VARCHAR2, v_objname IN
VARCHAR2) RETURN VARCHAR2 AS
     con VARCHAR2(200);
  BEGIN
     con := 'deptno = 20';
     RETURN con;
  END emp_policy_fn;
  /
  SELECT * FROM EMP;
```

Assuming the default edition is ORA$BASE, which is correct?

A. The query will return records pertaining to department 30 with SAL and COMM values displayed as NULL.

B. The query will return records pertaining to department 20.

C. The query will return records pertaining to department 20 with SAL and COMM values displayed as NULL.

D. The query will return records pertaining to department 30.

ANSWER: A

QUESTION NO: 6

Examine this function body:

```
BEGIN
  SELECT hire_date INTO date_hired FROM employees WHERE employee_id = emp_id;
  RETURN TO_CHAR (date_hired);
END;
```

Which two headers will allow this function to compile successfully and take advantage of both invoker's rights and function result caching? (Choose two.)

A. CREATE FUNCTION get_hire_date (emp_id NUMBER) RETURN VARCHAR2
RESULT_CACHE RELIES_ON (departments)
AUTHID CURRENT_USER
IS date_hired DATE;

B. CREATE FUNCTION get_hire_date (emp_id NUMBER) RETURN VARCHAR2
RESULT_CACHE
AUTHID CURRENT_USER
IS date_hired DATE;

C. CREATE FUNCTION get_hire_date (emp_id NUMBER) RETURN VARCHAR2
RESULT_CACHE
AUTHID DEFINER
IS date_hired DATE;

D. CREATE FUNCTION get_hire_date (emp_id NUMBER) RETURN VARCHAR2
RESULT_CACHE RELIES_ON (employees)
AUTHID CURRENT_USER
IS date_hired DATE;

E. CREATE FUNCTION get_hire_date (emp_id NUMBER) RETURN VARCHAR2
AUTHID DEFINER
IS date_hired DATE;

ANSWER: B D

QUESTION NO: 7

Examine this code:

```
CREATE PACKAGE measure IS
   SUBTYPE m_val IS NUMBER(10, 2);
   PROCEDURE calc_total (length IN m_val, extra_len IN OUT m_val, tot_length OUT
m_val);
END measure;

CREATE PACKAGE BODY measure IS
   PROCEDURE calc_total (length IN m_val, extra_len IN OUT m_val, tot_length OUT
m_val) IS
      BEGIN
         DBMS_OUTPUT.PUT_LINE ('LENGTH : ' || length);
         DBMS_OUTPUT.PUT_LINE ('EXTRA_LEN : ' || extra_len);
         tot_length := (length + extra_len);
      END;
END measure;

DECLARE
   total_len1 NUMBER;
   extra_1 measure.m_val;
BEGIN
   extra_1 := 15.247;
   measure.calc_total (100.1254, extra_1, total_len1);
   DBMS_OUTPUT.PUT_LINE ('TOTAL LENGTH : ' || total_len1);
END;
```

What will be the outcome with SERVEROUTPUT enabled?

A. The PL/SQL block will fail with a runtime exception.

B. LENGTH : 100.13 EXTRA_LEN : 15.25
TOTAL LENGTH : 115.38

C. LENGTH : 100.1254 EXTRA_LEN : 15.247
TOTAL LENGTH : 115.37

D. LENGTH : 100.1254 EXTRA_LEN : 15.25
TOTAL LENGTH : 115.38

E. LENGTH : 100.1254 EXTRA_LEN : 15.25
TOTAL LENGTH : 115.3754

ANSWER: A

QUESTION NO: 8

You are logged on to the SCOTT schema and the schema has EMP and DEPT tables already created.

Examine this PL/SQL procedure:

```
CREATE PROCEDURE get_tab_row_count (p_table_name IN VARCHAR2) AS
   l_sql VARCHAR2(200);
   l_count NUMBER;
BEGIN
   l_sql := 'SELECT COUNT(*) FROM ' || DBMS_ASSERT.SQL_OBJECT_NAME (p_table_name);
   EXECUTE IMMEDIATE l_sql INTO l_count;
   DBMS_OUTPUT.PUT_LINE ('l_count = ' || l_count);
END;
/
```

Which PL/SQL block will raise an exception?

A. EXEC get_tab_row_count (‘emp’);

B. EXEC get_tab_row_count (‘SCOTT.EMP’);

C. EXEC get_tab_row_count (‘ “EMP” ’);

D. EXEC get_tab_row_count (‘DEPT’);

E. EXEC get_tab_row_count (‘DEPT, EMP’)

ANSWER: E

QUESTION NO: 9

With SERVEROUTPUT enabled, you successfully create the package YEARLY_LIST:

```
CREATE PACKAGE yearly_list IS
    TYPE list1 IS TABLE OF VARCHAR2 (20) INDEX BY PLS_INTEGER;
    FUNCTION init_list1 RETURN list1;
END yearly_list;
/

CREATE PACKAGE BODY yearly_list IS
    FUNCTION init_list1 RETURN list1 IS
      create_list list1;
    BEGIN
      create_list(1) := 'Jan';
      create_list(3) := 'Feb';
      create_list(6) := 'Mar';
      create_list(8) := 'Apr';
      RETURN create_list;
    END init_list1;
END yearly_list;
/
```

Examine this code:

```
1 DECLARE
2     v_yrl yearly_list.create_list ();
3     location NUMBER :=1;
4 BEGIN
5     WHILE location IS NOT NULL LOOP
6         DBMS_PUTPUT.PUT_LINE (v(yrl (location) );
7         location := v_yrl.NEXT;
8     END LOOP;
9 END;
10 /
```

You want to display the contents of CREATE_LIST.

Which two lines need to be corrected in the PL/SQL block? (Choose two.)

A. Line 2

B. Line 3

C. Line 5

D. Line 6

E. Line 7

ANSWER: A E

QUESTION NO: 10

Examine this code:

```
CREATE PACKAGE pkg AS
    TYPE tab_typ IS TABLE OF VARHCAR2 (10) INDEX BY VARCHAR2;
    FUNCTION tab_end (p_tab IN tab_typ) RETURN tab_typ;
END pkg;
/
CREATE PACKAGE BODY pkg AS
   FUNCTION tab_end (p_tab IN tab_typ) RETURN tab-typ IS
   BEGIN
       RETURN p_tab.LAST;
   END;
END pkg;
/
DECLARE
   1_stmt VARCHAR2 (100);
   1_list pkg.tab_typ;
   1_result VARCHAR2 (10);
BEGIN
   1_list (1) := 'MONDAY';
   1_list (2) := 'TUESDAY';
   1_stmt := 'SELECT pkg.tab_end (:1_list) INTO :1_result FROM dual';
   EXECUTE IMMEDIATE 1_stmt INTO 1_result USING 1_list;
END;
```

Which two corrections must be applied for this anonymous block to execute successfully? (Choose two.)

A. Change RETURN p_tab.LAST to RETURN p_tab.COUNT.

B. Declare the collection type inside the function.

C. Declare the collection type at the schema level instead of the package.

D. Define the function as stand-alone instead of in a package body.

E. Change the INDEX BY clause from VARCHAR2 to PLS_INTEGER.

F. Modify the function return type to return a scalar, VARCHAR2.

ANSWER: E F