Data Dictionary
VB.NET Calling Stored Procedures and Functions

Tutorial Tasks

1. Consider this Anonymous Block:

```
1   DECLARE
2     rv_refcur SYS_REFCURSOR;
4     emp_details employee%ROWTYPE;
5   BEGIN
6     OPEN rv_refcur FOR SELECT * FROM employee;
7     LOOP FETCH rv_refcur INTO emp_details;
8       EXIT WHEN rv_refcur%NOTFOUND ;
9       DBMS_OUTPUT.PUT_LINE(emp_details.empname);
10    END LOOP;
11    CLOSE rv_refcur ;
12  END;
```

a) In regard to line 2, what is **rv_refcur?**
  ⇨ **It is a variable referenced to a cursor. More specifically, SYS_REFCURSOR is a predefined weak typed REF CURSOR that can be used to pass the result of a query between PL/SQL programs. Meaning that they can be use to get the result of multiple SELECT statement.**

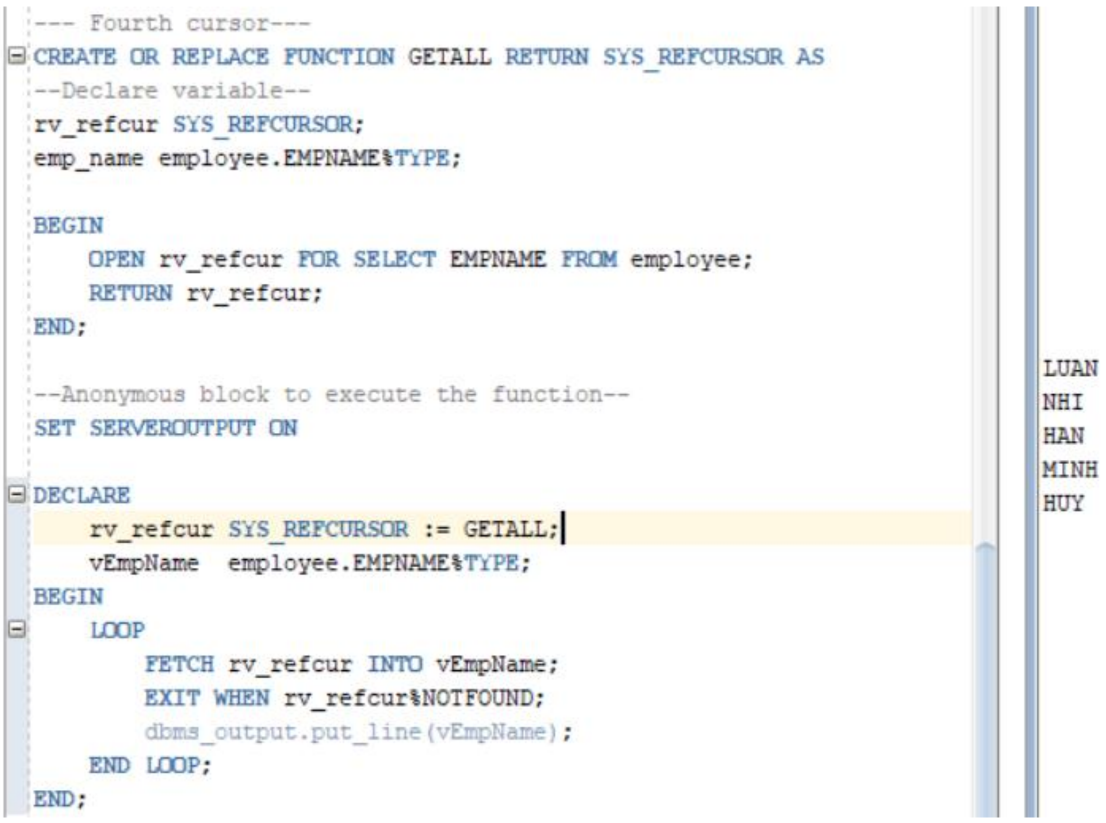
b) In regard to line 4, what is **emp_details?**
  ⇨ **It is a table-based record that will later be used to store result of rv_refcur.**

c) Describe what occurs as lines 5-12 are executed
  ⇨ **The cursor will fetch everything from table EMPLOYEE and the program will print each employee's name using emp_details record until there is nothing left in rv_refcur. After that we will close the cursor and end or program.**

2. Consider this Stored Function:

```
CREATE FUNCTION GETALL RETURN SYS_REFCURSOR AS
  rv_refcur SYS_REFCURSOR;
  emp_name employee.name%TYPE;
BEGIN
OPEN rv_refcur FOR SELECT emp_name FROM employee;
RETURN rv_refcur ;
END;
```

Write an Anonymous block that displays every name in the cursor returned by GETALL

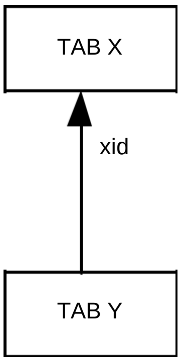


```
--- Fourth cursor---
CREATE OR REPLACE FUNCTION GETALL RETURN SYS_REFCURSOR AS
--Declare variable--
rv_refcur SYS_REFCURSOR;
emp_name employee.EMPNAME%TYPE;

BEGIN
    OPEN rv_refcur FOR SELECT EMPNAME FROM employee;
    RETURN rv_refcur;
END;

--Anonymous block to execute the function--
SET SERVEROUTPUT ON

DECLARE
    rv_refcur SYS_REFCURSOR := GETALL;
    vEmpName  employee.EMPNAME%TYPE;
BEGIN
    LOOP
        FETCH rv_refcur INTO vEmpName;
        EXIT WHEN rv_refcur%NOTFOUND;
        dbms_output.put_line(vEmpName);
    END LOOP;
END;
```

```
LUAN
NHI
HAN
MINH
HUY
```

3. A database has two tables TABX and TABY.

```
CREATE TABLE TABX (
xid       NUMBER         PRIMARY KEY,
xname     VARCHAR2(10) );
CREATE TABLE TABY (
yid       NUMBER         PRIMARY KEY,
yname     VARCHAR2(10) ,
xid       NUMBER,
FOREIGN KEY  (xid)   REFERENCES TABX );
```



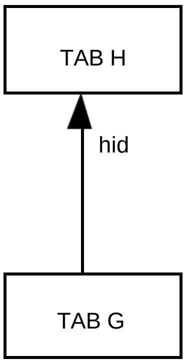
Note:
The foreign key in this Oracle example uses the **default** ON DELETE NO ACTION
Notice that Oracle does **not** explicitly state 'ON DELETE NO ACTION'.
In fact if you attempt to explicitly add 'ON DELETE NO ACTION' to the foreign key constraint, then it will generate a compiler error.

Assume that **every** row in TABX has one or more child rows in TABY

- True or False? The Parent row in TABX may be deleted. => **False**
- True or False? Any child of a parent row in TABX may be deleted => **True**
- True or False? When a parent in TABX is deleted, each child is automatically deleted => **False**
- True or False? When a parent in TABX is deleted, each child's FK is set to NULL => **False**
- True or False? When a child in TABY is deleted, the parent in TABX is automatically deleted => **False**

4. A database has two tables TABG and TABH.

```
CREATE TABLE TABH (
hid       NUMBER        PRIMARY KEY,
hname     VARCHAR2(10) );
Create Table TABG (
gid       NUMBER        PRIMARY KEY,
gname     VARCHAR2(10) ,
hid       NUMBER,
FOREIGN KEY  (hid)   REFERENCES TabH ON DELETE SET NULL );
```

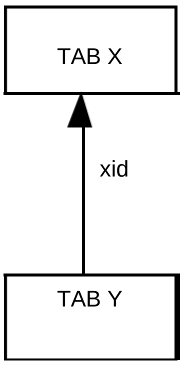


Assume that every row in TABH has one or more child rows in TABG

- True or False? The Parent row in TABH may be deleted. => **True**
- True or False? Any child of a parent row in TABH may be deleted => **True**
- True or False? When a parent in TABH is deleted, each child is automatically deleted => **False**
- True or False? When a parent in TABH is deleted, each child's FK is set to NULL => **True**
- True or False? When a child in TABG is deleted, the parent in TABH is automatically deleted

=> **False**

5. A database has the same two tables TABX and TABY.

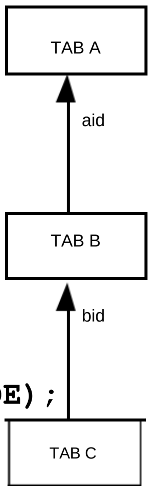


```
CREATE TABLE TABX (
xid      NUMBER        PRIMARY KEY,
xname    VARCHAR2(10) );
CREATE TABLE TABY (
yid      NUMBER        PRIMARY KEY,
yname    VARCHAR2(10) ,
xid      NUMBER,
FOREIGN KEY  (xid)   REFERENCES TabX ON DELETE CASCADE);
```

Assume that every row in TABX has one or more child rows in TABY

- True or False? The Parent row in TABX may be deleted. => **True**
- True or False? Any child of a parent row in TABX may be deleted => **True**
- True or False? When a parent in TABX is deleted, each child is automatically deleted => **True**
- True or False? When a parent in TABX is deleted, each child's FK is set to NULL => **False**
- True or False? When a child in TABY is deleted, the parent in TABX is automatically deleted => **False**

6. A database has the same three tables TABA , TABB and TABC.



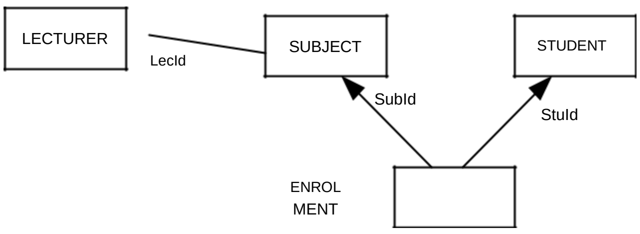
```
CREATE TABLE TABA (
aid      NUMBER        PRIMARY KEY,

aname    VARCHAR2(10) );
CREATE TABLE TABB (
bid      NUMBER        PRIMARY KEY,
bname    VARCHAR2(10) ,
aid      NUMBER,
FOREIGN KEY  (aid)   REFERENCES TABA ON DELETE CASCADE);
CREATE TABLE TABC (
cid      NUMBER        PRIMARY KEY,
cname    VARCHAR2(10) ,
bid      NUMBER,
FOREIGN KEY  (bid)   REFERENCES TABB);
INSERT INTO TABA (aid, aname) VALUES (1,'A1');
INSERT INTO TABA(aid, aname) VALUES (2,'A2');
INSERT INTO TABB(bid, bname, aid) VALUES (5,'B5', 1);
INSERT INTO TABB(bid, bname, aid) VALUES (6,'B6', 1);
INSERT INTO TABC(cid, cname, bid) VALUES (8,'C8', 5);
INSERT INTO TABC(cid, cname, bid) VALUES (9,'C9', 5);
```

- T or F. The TABC row with id C8 may be deleted. If true, what is the effect on other rows
  ⇨ **True, no effect on other rows.**
- T or F. The TABC row with id C9 may be deleted. If true, what is the effect on other rows
  ⇨ **True, no effect on other rows.**
- T or F. The TABB row with id B5 may be deleted. If true, what is the effect on other rows
  ⇨ **False, B5 can't be deleted.**
- T or F. The TABB row with id B6 may be deleted. If true, what is the effect on other rows

- ⇨ **True, no effect on other rows.**

- T or F. The TABA row with id A1 may be deleted. If true, what is the effect on other rows
  - ⇨ **False, A1 can't be deleted.**
- T or F. The TABA row with id A2 may be deleted. If true, what is the effect on other rows
  - ⇨ **True, no effect on other rows.**

7. Consider the following network diagram and DDL



```
CREATE TABLE lecturer (
      lecid INTEGER PRIMARY KEY ,
      lecname VARCHAR2(30) );
```

```
Insert into lecturer values (1,'Dave the lecturer');
Insert into lecturer values (2,'Sue the lecturer');
Insert into lecturer values (3,'Hana the lecturer');
```

```
CREATE TABLE subject (
      subid INTEGER PRIMARY
      KEY, subcode VARCHAR2
      (10), subname VARCHAR2
      (30), lecid INTEGER,
      FOREIGN KEY (lecid) REFERENCES
      lecturer ON DELETE SET NULL  );
```

```
Insert into subject values (101,'HIT1401','IBIS',1);
Insert into subject values (202,'HIT1402','DAD',2);
Insert into subject values (303,'HIT1403','ICTE',3);
Insert into subject values (404,'HIT1404','IPROG',1);
```

```
CREATE TABLE student (
      stuid INTEGER PRIMARY KEY,
      stuname VARCHAR2 (30) );
```

```
Insert into student values (551,'Jim Student');
Insert into student values (552,'Tom Student');
Insert into student values (553,'Jane Student');
Insert into student values (554,'Emma Student');
```

```
CREATE TABLE enrolment (
      enrolid INTEGER PRIMARY
      KEY, stuid INTEGER,
      subid INTEGER,
      FOREIGN KEY (stuid) REFERENCES
      student ON DELETE CASCADE,
      FOREIGN KEY (subid) REFERENCES subject);
```

```
Insert into enrolment values (901, 551, 101);
Insert into enrolment values (902, 551, 202);
Insert into enrolment values (903, 553, 101);
Insert into enrolment values (904, 554, 202);
Insert into enrolment values (905, 553, 303);
```

What effect will the following statements have?

a. DELETE FROM Enrolment WHERE enrolId = 905;

⇨ **The row where enrolid = 905 will be deleted, no effect on other rows.**

b. DELETE FROM Subject WHERE subId = 404;

⇨ **The row where subid = 404 will be deleted, no effect on other rows.**

c. DELETE FROM Subject WHERE subId = 101;

⇨ **SubID 101 can't be deleted.**

d. DELETE FROM Lecturer WHERE lecId = 1;

⇨ **The row where lecid = 1 in Lecturer table will be deleted, the row where lecid = 1 in Subject table will be set to NULL.**

e. DELETE FROM Student WHERE stuname = 'Tom Student';

⇨ **The row where stuname = 'Tom Student' in Enrolment and Student Table will be deleted.**

f. DELETE FROM Student WHERE stuId = 553;

⇨ **The row where stuid = '553' in Enrolment and Student Table will be deleted.**

8. The Data Dictionary is sometime referred to a Meta Data repository.
What sort of information is stored in a Data Dictionary or a Meta Data repository
It may contain the following:
+ Names and definitions of data objects.
+ Properties of data elements (data types, size, nullability,…).
+ Entity-Relationship
+ Users and Users' constraints.

9. Which statement is correct:
   a. "An Oracle data dictionary is a set of tables"
   b. "An Oracle data dictionary is a set of views"
   c. "An Oracle data dictionary is a set of tables and views".

- ⇨ **It is a set of tables.**

10. Can a database programmer or use directly update the Data Dictionary?
    - ⇨ **No we can not**
11. How is the Data Dictionary updated?
    - ⇨ **The Data Dictionary is automatically updated when we create new table.**
12. How will the result differ in these two statements:

```
 SELECT object_name, object_type FROM USER_OBJECTS; 
SELECT object_name, object_type FROM ALL_OBJECTS;
```

- ⇨ **The first will shows all current user objects**
- ⇨ **The second will show all current objects and the object the users have access to.**

13. Consider the statement:
    - `SELECT table_name, num_rows  FROM  USER_TABLES;`
    - `SELECT table_name, num_rows  FROM  ALL_TABLES;`

    Will you see the same result set for both?
    - ⇨ **No the first will show all the data that the user owns, the second will show all the data that the user owns and the data from tables that the user has access to.**
14. Can you query the DD for all the tables in ALL student accounts? Why?/Why not?
    - ⇨ **Yes we can use query to access data from one or many tables.**
15. Suppose that you type "SELECT * FROM s1234567.DEPT; " (where s1234567 is a DB user account)
    How does the DBMS determine
    - If you have permission to access this table?
    - Which columns are to be displayed?

      => **DBMS will first check if I have the permission to read this account's tables, if I do then I will be able to view it. All the tables that s1234567 owns can be viewed.**
16. Consider these 4 data dictionary tables:
    TABLES, TAB_COLUMNS, CONSTRAINTS and CONS_COLUMNS Imagine

    that the data values within the data dictionary looks like this:

    TABLES

Table Name
Student

    TAB_COLUMNS

Column Name	TableName	DataType
StuId	Student	Number
StuName	Student	Varchar2(100)

    CONSTRAINTS

Constraint Name	Constraint Type	TableName	Condition
PK_STUDENT_STUID	P		
NN_STUDENT_STUNAME	N		

    CONS_COLUMNS

Column Name	TableName	Constraint Name
StuId	Student	PK_STUDENT_STUID
StuName	Student	NN_STUDENT_STUNAME

    Re-construct one Create Table statement that could have been used to create these Data Dictionary values.

    CREATE TABLE STUDENT(

    StuId Number,

    StuName Varchar2(100) CONSTRAINT NN_STUDENT__NAME NOT NULL,
CONSTRAINT PK_STUDENT_STUID PRIMARY KEY (StuId));

17. Consider these 4 data dictionary tables:
TABLES, TAB_COLUMNS, CONSTRAINTS and CONS_COLUMNS Imagine that the data values within the data dictionary looks like this:

TABLES

Table Name
Branch
Employee

TAB_COLUMNS

Column Name	TableName	DataType
BranchId	Branch	Number
BranchName	Branch	Varchar2(20)
EmpId	Employee	Number
Firstname	Employee	Varchar2(50)
Surname	Employee	Varchar2(50)
Salary	Employee	Number
BranchId	Employee	Number

CONSTRAINTS

Constraint Name	Constraint Type	TableName	Condition
PK_BRANCH	P		
PK_EMPLOYEE	P		
FK_EMPLOYEE_BRANCHID	F	BRANCH	
CC_EMPLOYEE_SALARY	F		(Salary > 0)

CONS_COLUMNS

Column Name	TableName	Constraint Name
BranchId	Branch	PK_BRANCH
EmpId	Employee	PK_EMPLOYEE
BranchId	Employee	FK_EMPLOYEE_BRANCHID
Salary	Employee	CC_EMPLOYEE_SALARY

Re-construct two Create Table statements that could have been used to create these Data Dictionary values.

18. What additional rows would be made to the Data Dictionary if the following Alter Statements are executed
ALTER TABLE Employee
ADD CONSTRAINT uc_employee_name UNIQUE (Firstname, Surname);

   ⇨ **3 Additional rows would be added, uc_employee_name in CONSTRAINTS, Firstname and Surname in CONS_COLUMNS**

19. If you run the following DDL script:

```
CREATE TABLE                                          EMPLOYEE
                      (EMPID   NUMBER,
                       NAME    NUMBER,
                       GENDER  VARCHAR(1),
                       CONSTRAINT PK_EMPLOYEE PRIMARY KEY (EMPID),
                       CONSTRAINT NN_EMPLOYEE_GENDER NOT NULL (GENDER),
                       CONSTRAINT CC_EMPLOYEE_GENDER CHECK (GENDER IN ('M','F')) );
```

List the values in each of these data dictionary objects:
TABLES, TAB_COLUMNS, CONSTRAINTS and CONS_COLUMNS

TABLES

TABLE NAME
EMPLOYEE

Tab_COLUMNS

COLUMN NAME	TABLE NAME	DATA TYPE
EMPID	**EMPLOYEE**	**NUMBER**
NAME	**EMPLOYEE**	**NUMBER**
GENDER	**EMPLOYEE**	**VARCHAR(1)**

CONSTRAINTS

CONSTRAINTS NAME	CONSTRAINT TYPE	TABlE NAME	CONDITION
PK_EMPLOYEE	**P**		
NN_EMPLOYEE GENDER			
CC_EMPLOYEE GENDER			**GENDER IN (‘M’,’F’)**

CONS_COLUMNS

COLUMN NAME	TABLE NAME	CONSTRAINT NAME
EMPID	**EMPLOYEE**	**PK_EMPLOYEE**
GENDER	**EMPLOYEE**	**NN_EMPLOYEE GENDER**
GENDER	**EMPLOYEE**	**CC_EMPLOYEE GENDER**

Lab Tasks:

Continue with assignment work