Exercise #1: Simple dynamic cursor

1. In an anonymous PL/SQL block, define a string variable named *tablename* and initialize it with the name of any table in your account. Write the PL/SQL code that displays this table's schema (i.e., list of attributes) with respect to format `TABLE_NAME (ATT1, ATT2, …)`. Table attributes are recorded in system view `USER_TAB_COLUMNS (TABLE_NAME, COLUMN_NAME, …)`.

2. Transform the anonymous PL/SQL block into a stored procedure named *schema* that inputs the name of the table to process.

Exercise #2: Application

Let be three product catalogs (tables) that we want to integrate into a single one. All catalogs do not exactly bear the same structure. They are listed in table META, which also specifies how their data may be transformed. You may copy all tables from DARMONT.C1, DARMONT.C2, etc.

C1

product_id	*product_name*	*product_price*
1	COMPUTER	€799.90
2	MONITOR	€349.9

C3

pname	*pprice*
mouse	$29,9
webcam	$19,9

C2

Id	*price*	*name*
10	€299.90	Printer
20	€149.9	Scanner

META

table_name	*trans_code*
C1	NULL
C2	CAP
C3	CAP+CUR

Write an anonymous PL/SQL block that:

- dynamically creates a new table named C_ALL (pid, pname, pprice);
- scans table META to retrieve all catalog names (i.e., C1, C2 and C3 here);
- dynamically retrieve attribute names from each catalog schema looking like "NAME" and "PRICE", respectively;
- dynamically load data from each catalog into C_ALL with the help of previously found attribute names;
- generate the ID in C_ALL automatically (discard other existing IDs);
- when loading data, apply suitable transformations:
  - CAP means product name must be put in large caps,
  - CUR means product price must be converted into Euros (with €1 = $1.15).

Pseudocode:

```
Create table C_ALL (after 1st execution, drop table C_ALL beforehand)

For each row in table META loop
    Select attribute name in table USER_TAB_COLUMNS that looks like NAME
        in current table; store it into variable att_name
    Select attribute name in table USER_TAB_COLUMNS that looks like PRICE
        in current table; store it into variable att_price
    Open dynamic cursor: Select att_name and att_price from current table
    For each row in dynamic cursor loop
        Store row in variables name and price
        Increment catalog ID
        If META transformation code looks like CAP then
            name ← UPPER(name)
        End if
        If META transformation code looks like CUR then
            price ← price / exchange rate
        End if
        Insert into table C_ALL values ID, name and price
    End loop
    Close dynamic cursor
End loop
Commit transaction
```