TABLE1

A_1	B_1	C_1	A_2	B_2	C_2	...	A_n
D	E	F	G	H	I		1
D	E	F	G	H	I		2
D	E	F	G	H	I		n
E	F	G	H	I	D		1
E	F	G	H	I	D		2
E	F	G	H	I	D		n
...							

This is the original table, which is filled without my influence – so I have to take it the way it is. It contains a very large set of data, which we need to create rules for our application. The rules are composed of:
the X_1 attributes + A_n
the X_2 attributes + A_n
until the X_n-1 attributes + A_n

The approach my former colleague took, was to create a cursor with all relevant attributes and ALL columns and extract the X_n attributes separately from each row and pair it with the A_n attribute and put it into another table. In the beginning, TABLE1 was small and everything worked fine. But now, when we generate the rules, we have to go through a huge table (which is NOT normalized) and it takes far to long.

So what I'm trying to do is the following:

```
cursor myCursor1 is
select distinct
      A_1, B_1, C_1, A_2, B_2, C_2, … C_n-1
from table1
where criterion = criterion_n;
```

From each row I extract these distinct A to C triples and put them into a new table with a rule number and some other relevant stuff :

TABLE2

Rule#	Attribute1	Attribute2	Attribute3
011	D	E	F
012	G	H	I
021	E	F	G
022	H	I	D
...			

Then I declare another cursor within the first one

```
cursor myCursor2 is
      select t1.A_n
      from table1 t1
      where t1.A_1 = myCursor2.A_1
      AND t1.A_2 = myCursor2.A_2
      AND t1.A_3 = myCursor2.A_3;
```

and select all A_n attributes which exist for the current row of the first cursor and put them into another table:

TABLE3

Rule#	A_n
011	1
011	2
011	n
012	1
012	2
012	n
02	n
021	1
021	2
021	n
022	1
022	2
022	n
...	

All this works fine, but now comes the point where I very much need your help:
Depending on the criterion used for the first cursor (`where criterion = criterion_n`) the A_n attribute can be at A_3, A_4 or A_5 and I cannot exclude the possibility that some day it will be A_2 or A_6. So what I want is a more generic approach to the whole problem. And I wanted to make some use of the 'horrible' fact that the column names in TABLE1 are numbered – something like that:

```
determine criterion n
then do all the above described with attributes 1 to n-1 and use A_n
```

And my idea was to use dynamic SQL and pass different values to the cursors, depending on the criterion – but I didn't manage to do that…

Something like:

```
begin
      i := criterion;
      attributeStringA    := 'myCursor1.A_';
      attributeStringB    := 'myCursor1.B_';
      attributeStringC    := 'myCursor1.C_';

        loop
          exit when i<1;
            sql_stmt := 'insert into table2 values (:1, :2, :3)';
            i_string := to_char( i, '9');
            A_i := attributeStringA || i_string;
            B_i := attributeStringB || i_string;
            C_i := attributeStringC || i_string;
            my_kennzeichen := cast(internes_knz_x as %type);
            execute immediate sql_stmt using A_i, B_i, C_i;
          i := i-1;
        end loop;
end;
```

But this doesn't work, because the respective attributes in TABLE2 expect a varchar2 and thus the strings 'A_i', 'B_i', 'C_i' are written into TABLE2 instead of the values represented by the variables – even though I didn't put them into quotes.

I'm very sorry for posting all this stuff, even though I have only a basic question about passing variable values to a dynamic SQL statement, but I thought it's important for you to know why I'm doing what I'm doing ;-) And maybe you see a better/easier way of doing all this.

Thank you very much in advance!
Bianca