PLSQL Tuning: Bind Variables and execute immediate

With this blog post I am trying to highlight the Performance benefits of Bind Variables and How to use bind variables with execute immediate. I mostly do not like to have theory in my blog posts as it is already there perfectly in Oracle Documentation but here I am trying to give a little background.

First of all what is bind variable:

A bind variable is basically a place-holder SQL statement which is replaced by actual value when the SQL statement is executed. Bind Variables helps application to send exactly the same SQL to Oracle every time. Bind variable can highly improve performance by avoiding hard parsing. It also helps Oracle to lower Shared Pool Memory consumption and also helpful to avoid SQL injection.

Now about Hard Parsing and Soft Parsing

For each SQL submitted, Oracle picks the execution plan from shared pool if same exact SQL already exists there, which is called soft parsing. If the SQL submitted is not found in Shared Pool, Oracle has to do the hard parsing which is SQL statement needs to be checked for syntax and semantics errors, and generating various execution plans to find and select optimal one. Hard parsing is very CPU intensive.

Now my favourite part, lets code and check performance benefits of bind variables.

Prepare data for tesing the performance.

```
SQL> create table test_bind as
  2  select level col from dual
  3  connect by level <= 99999;
Table created.

SQL> commit;
Commit complete.

SQL> create index test_indx on test_bind(col);
Index created.

SQL> select count(*) from test_bind;
  COUNT(*)
----------
     99999
```

All good, we have created a table with 99999 records and we have an index on it. Lets run plsql block executing 99999 queries without bind variable.

```
SQL> set timing on
SQL> declare
  2     a number;
  3     i number;
```

```
  4     mysql varchar2(100);
  5  begin
  6    i := 1;
  7    while i<=99999
  8    loop
  9      mysql := 'select * from test_bind where col=' || i;
 10      execute immediate mysql
 11         into a;
 12      i := i+1;
 13    end loop;
 14  end;
 15  /
PL/SQL procedure successfully completed.
Elapsed: 00:27:15.55
```

99999 unique queries executed in more than 27 minutes, almost 60 queries per second.

Lets now try same code with bind variables on the same data

```
SQL> set timing on
SQL> declare
  2     a number;
  3     i number;
  4     mysql varchar2(100);
  5  begin
  6    i := 1;
  7    while i<=99999
  8    loop
  9       mysql := 'select * from test_bind where col=:1';
 10       execute immediate mysql
 11           into a
 12          using i;
 13       i := i+1;
 14     end loop;
 15  end;
 16  /
PL/SQL procedure successfully completed.
Elapsed: 00:00:05.03
```

WOW, It got executed in just 5 seconds, what a performance gain almost 20000 queries in a second. We just changed is just used bind variables and found 95% performance gain. Hard Parsing seems to be a really resource/CPU consuming task.