

Automatsko generisanje test podataka

Ivan Korać

U poslu se često dešavaju situacije da treba da ugradite neku business logiku u stored package, a da često dobijete samo opis tabela bez flat file-ova sa test podacima. Kako god bilo isprogramirana, logika se mora testirati na nekim podacima.

Izuzetno je naporno kreirati insert statement-e za punjenje test podataka, prvo zato što može da se radi o milionima slogova - potrebnih za testiranje performansi, a drugo što test podaci najčešće treba da poštuju određene domene i pravila (constraints), npr. foreign key constraint.

Zato je više nego potrebno imati pri ruci različite dynamic procedure, koje bi zavisno od tipa kolone u tabeli i/ili constrainta insertovale željeni broj slogova u tabelu.

Ovde će biti navedeno nekoliko ideja proizašlih iz konsultacija sa Thomasom Kytom (autor "najjače" ikad izdate Oracle knjige: Expert one-on-one).

Prvi primer je procedura koja klonira već postojeću tabelu (kopira strukturu) i puni je sa željenim brojem slogova respektujući tip kolone. Umesto dbms_random package-a (proverite da li je dbms_random instaliran u vašoj bazi), možete koristiti svoje vrednosti.

Primer 1

```
> create or replace procedure clone( p_tname in varchar2, p_records in number )
2  authid current_user
3  as
4  l_insert long;
5  l_rows  number default 0;
6  begin
7
8  execute immediate 'create table clone_' || p_tname ||
9      ' as select * from ' || p_tname ||
10     ' where 1=0';
11
12  l_insert := 'insert into clone_' || p_tname ||
13     ' select ';
14
15  for x in ( select data_type, data_length,
16     nvl(rpad('9',data_precision,'9')/power(10,data_scale),999999999) maxval
17     from user_tab_columns
18     where table_name = 'CLONE_' || upper(p_tname)
19     order by column_id )
20  loop
21     if ( x.data_type in ('NUMBER', 'FLOAT' ))
22     then
```



```
23     l_insert := l_insert || 'dbms_random.value(1, || x.maxval || ),';
24     elsif ( x.data_type = 'DATE' )
25     then
26         l_insert := l_insert || 'sysdate+dbms_random.value+dbms_random.value(1,1000),';
27     else
28         l_insert := l_insert || 'dbms_random.string("A", || x.data_length || ),';
29     end if;
30 end loop;
31 l_insert := rtrim(l_insert,',') || ' from all_objects where rownum <= :n';
32
33 loop
34     execute immediate l_insert using p_records - l_rows;
35     l_rows := l_rows + sql%rowcount;
36     exit when ( l_rows >= p_records );
37 end loop;
38 end;
39 /
```

Procedure created.

```
> exec clone( 'emp', 5 );
```

PL/SQL procedure successfully completed.

```
> select * from clone_emp;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
4923	BrCyhfdHaQ	mGSqkWMvY	8302	07-SEP-02	2765.67	89231.32	27
323	ImuBZCYDrt	TdjoFlvYE	9613	11-AUG-03	82158.44	34478.25	89
8773	jnTPtKkchC	KzUezmTTL	8432	20-MAY-04	29909.7	23860.67	24
7374	aETDfeptSS	ObVBgtNAP	9033	28-MAR-03	34012.77	92188.64	45
4530	MKGvauUMOQ	GXwcRjGUn	51	07-AUG-04	84510.9	93629.66	79

Da bi se pri generisanju podataka, poštovalo pravilo foreign key-a, navodimo ideju kako slučajno izabrati proizvoljno veliki set vrednosti iz već postojećeg. Npr. u tabelu emp, polje deptno je spoljni ključ iz tabele dept.

Ideja je da se vrednosti primarnog ključa iz tabele DEPT napune u plsql table type i koristi dbms_random da izindeksira slučajno izabrane vrednosti u nizu:

Primer 2

```
> declare
```

```
2     type numArray is table of number index by binary_integer;
```

```
3
```

```
4     l_deptnos numArray;
```

```
5 begin
```



```
6      select deptno bulk collect into l_deptnos from dept;
7
8      for i in 1 .. 5
9      loop
10         dbms_output.put_line( l_deptnos( dbms_random.value( 1, l_deptnos.count ) ));
11     end loop;
12 end;
13 /
```

PL/SQL procedure successfully completed.

Instaliranje DBMS_RANDOM package-a

Za instaliranje dbms_random package, potrebno je da se konektujete kao user SYS ili INTERNAL i da pokrenete sledece skriptove:

```
SVRMGR> connect internal
```

Connected.

```
SVRMGR> @utlraw
```

Statement processed.

```
SVRMGR> @prvtrawb.plb
```

Statement processed.

```
SVRMGR> @dbmsoctk
```

Statement processed.

Statement processed.

Statement processed.

Statement processed.

Statement processed.

Statement processed.

```
SVRMGR> @prvtocTk.plb
```

Statement processed.

Statement processed.

Statement processed.

```
SVRMGR> @dbmsrand
```

Statement processed.

Statement processed.

Statement processed.

Statement processed.

Statement processed.

```
SVRMGR>
```

Po instaliranju za “quick start”, potrebno je samo:



```
SVRMGR> select text from all_source
2 where name = 'DBMS_RANDOM'
3 and type = 'PACKAGE'
4 order by line;
```