Quick Introduction to SQL Translation Framework from Docs

Various client-side applications, designed to work with non-Oracle Databases, cannot be used with Oracle Databas without significant alterations. This is because SQL dialect varies among vendors of database technologies and different vendors use different syntaxes to express SQL queries and statements.

Starting with Oracle Database Release 12*c*, there is a new mechanism called SQL Translation Framework that translates the SQL statements of a client program from a foreign (non-Oracle) SQL dialect into the SQL dialect use the Oracle Database SQL compiler.

In addition to translating non-Oracle SQL statements, the SQL Translation Framework can also be used to substitut Oracle SQL statement with another Oracle statement to address a semantic or a performance issue. In this way, yo can address an application issue without patching the client application.

The SQL translation framework consists of the following two components:

- The SQL Translator
- The SQL Translation Profile

There were few words what is Translation Framework and how can be used in real life for example how to easy cor Sybase dialect to Oracle dialect without change code of application

```
select top 4 *  from
to
select * from emp wh
```

To do it we can use prepared by Oracle translation profiles for specific SQL dialect for example for Sybase.

SQL Translation Framework give us also much more powerful and intersting feature which is call Custom Translatic where we can implement on-the-fly translation from one SQL query to second one.

All we know the DBMS_Advanced_Rewrite which can do similar job, but using this package is more complicated ar have more restriction in real life.

OK so let's do some testing using Custom SQL Translation, we have databases called Stitch (as in popular cartoon

First me must do is to create translation profile and some translation which will be used in our tests

```
exec dbms_sql_trans

BEGIN
DBMS_SQL_TRANSLATOR
     profile_name
     sql_text
     translated_tex
END;



BEGIN
   DBMS_SQL_TRANSLA
     profile_name
     sql_text
     translated_tex
END;
```

So let's check how it's work on Stitch Database

```
2 alter session set ev
```

This must be done to simulate that we are remote client of database , without this translation won't be apply for the internal SQLPlus connected session.

```
1 select 888 from dual
2
3 11111
```

So let's check what we can see in SQL area , which SQL query is present Translated or Original ??

```
1  SQL_ID 7tr6y6c6ua5d
2  --------------------
3  select 11111 from d
4
5  Plan hash value: 13
6
7  --------------------
8  | Id  | Operation
9  --------------------
10 |   0 | SELECT STAT
11 |   1 |  FAST DUAL
12 --------------------
13
14 13 rows selected.
```

Next subject how it is organizing match pattern ?

```
1 Select 888        fr
2
3 888
```

So beware that matching must be exactly the same string as in DBMS_SQL_TRANSLATOR sql_text.

Ok so this translation will work but what if we try to translate DDL SQL

```
1 BEGIN
2    DBMS_SQL_TRANSLAT
3      profile_name
4      sql_text
5      translated_text
6 END;
```

alter table obj_test enable row;

Table dropped.

```
1 select * from obj_te
2 select * from obj_te
3               *
4 ERROR at line 1:
5 ORA-00942: table or
```

Ok so as you see we can do it and we must have in our mind that this technology and be real security problem in ca of Injection.

But let' s see how Oracle try to minimize this problem:

```
1 BEGIN
2    DBMS_SQL_TRANSLAT
3      profile_name
4      sql_text
```

```
select * from test2;

ORA-00900: invalid S
```

Ok. so we see that Oracle accept translation command to command so select to select ,

delete to delete etc.

So I wondering if we can cheat Oracle ?? I'm pretty sure that it is possible with no additional big effort , I got idea in seconds.

Let's do it select into select translation :

```
BEGIN
   DBMS_SQL_TRANSLAT
     profile_name
     sql_text
     translated_text
END;
```

So what is that mystery PURGEDATA ??

create or replace function PURGEDATA return number is

```
pragma autonomous_t

begin
 DELETE FROM object
 COMMIT;
 return 1;
 EXCEPTION WHEN OTH
 return 0;
end PURGEDATA;
/
```

So let run

```
select count(*) fro

95672 rows



select (*) from obj

select count(*) fro

0 rows
```

So it is working like a charm , so let's see what will tell us plan of this statement :

```
SQL_ID bwx3wf3a54p0
--------------------
select PURGEDATA fr

Plan hash value: 13

-------------------
| Id  | Operation |
-------------------
|   0 | SELECT STAT
|   1 |  FAST DUAL
-------------------
```

So as you see Translation Framework is great tool but it can be great hole to compromise our database in case of injection, so you can used it but please use it very careful.

Much more about this translation and other security threat will be covered in one of next post.