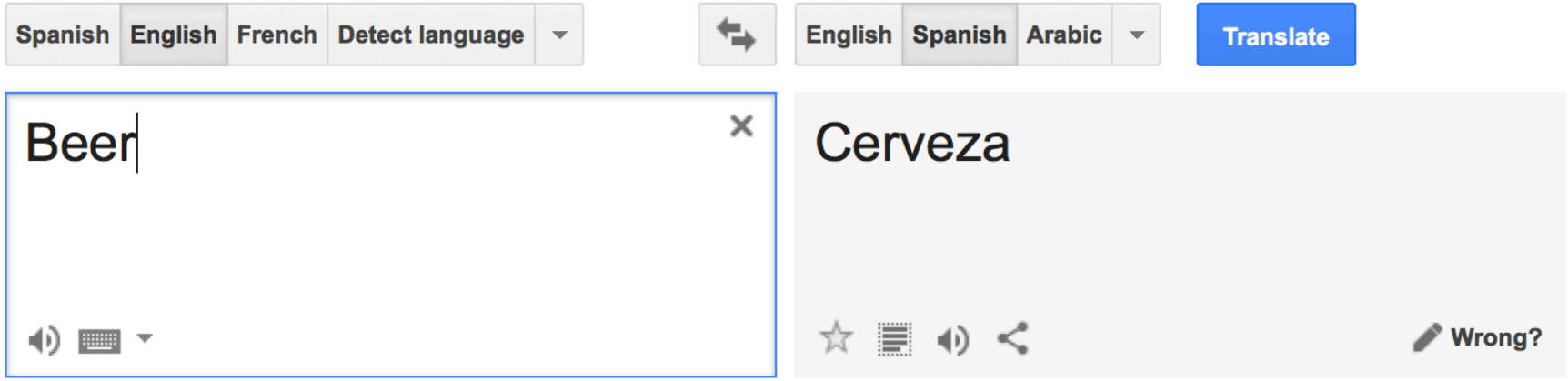
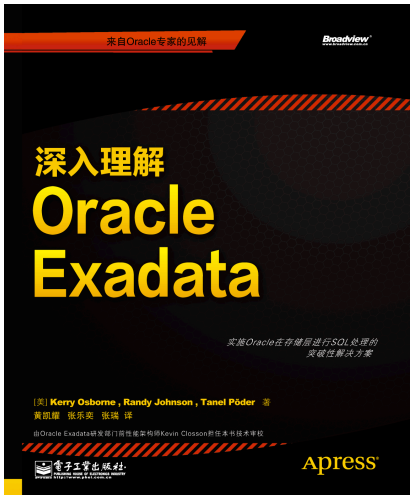
SQL Translation Framework



whoami –

Never Worked for Oracle
Worked with Oracle DB Since 1982 (V2)
Work for Enkitec
- Now a part of Accenture
I usually wear a ball cap
Haven't worn a necktie in a few decades
Decided long ago that it takes too long to get haircuts



Email: kerry.osborne@enkitec.com
Blog: kerryosborne.oracle-guy.com
Twitter: @KerryOracleGuy

OakTable.net

ORACLE ACE Director

What is SQL Translation Framework?

Feature of 12c release 1
Provides capability to transparently translate SQL
Came out of SQL Developer
Designed to translate non-Oracle syntax
But – provides ability to create your own translations ☺

Why Am I interested?

Because Sh*t Happens!
Performance issues occur
- bad plans
- bad SQL

It's often difficult or time consuming to make application changes
There are several mechanisms to change plans (hint based)

- Baselines
- SQL Profiles
- SQL Patches
- Outlines

Digression: Hint Based Mechanisms

Apply set of hints behind the scenes
They do not “lock” plans
- although they can appear to have that affect

Digression: Baselines vs SQL Profiles

SQL Profiles
- Designed to apply statistical fixes (fudge factors)
- Via SQL Tuning Advisor
- Uses opt_estimate hint
- However, they can be created manually with any hints

Baselines
- Designed to prevent performance regression
- Allow for multiple plans
- Evolution mechanism
- Know plan_hash_value

But sometimes the SQL has to change

Where hints won't work

```
select sum(distinct X)
from T1, T2
where T1.start_date < sysdate-7
and T2.dept = 'XYZ'


Blah Blah

   MERGE JOIN CARTESIAN

Blah Blah
```

SQL Translation Framework Coolness!

12.1 Migration Guide:

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

Basic Mechanics

1. Get Access to dbms_sql_translator
2. Create Translation Profile (create_profile)
3. Create SQL mappings (register_sql_translations)
4. Tell a session to use the Profile
5. Set the 10601 event

See Blog Entry:

http://kerryosborne.oracle-guy.com/2013/07/sql-translation-framework/

Tracing

exec DBMS_SQL_TRANSLATOR.SET_ATTRIBUTE
(’FOO’,'TRACE_TRANSLATION','TRUE');

. . .

SQL Translation Profile "SYS"."FOO": original SQL text
"select /* kso6 */ sum(pk_col) from kso.skew”

SQL Translation Profile "SYS"."FOO": translated SQL text "select /*
kso6 */ sum(pk_col), count(pk_col) from kso.skew"

Viewing Translations

DBA_SQL_TRANSLATIONS

```
SYS@gluent> @translations

OWNER      PROFILE_NA SQL_TEXT                       TRANSLATED_TEXT
---------- ---------- ------------------------------ ------------------------------
SYS        FOO        select /* kso6 */ sum(pk_col)  select /* kso6 */ sum(pk_col),
                      from kso.skew                  count(pk_col) from kso.skew


SYS        FOO        select max(col1) from kso.skew select distinct col4 from
                                                     kso.skew


SYS        FOO        select /*+ noparallel */       select /*+ noparallel */
                      a.col2, sum(a.col1) from       a.col2, sum(a.col1) from
                      kso.skew3 a, kso.skew b group  kso.skew3 a, kso.skew b where
                      by a.col2                      a.pk_col = b.pk_col group by
                                                     a.col2


SYS        FOO        select 5 from dual             delete from dual
```

Viewing Translations

V$MAPPED_SQL

```
SYS@gluent1> @mapped_sql
Enter value for sql_text:
Enter value for mapped_sql_text:

SQL_ID        SQL_TEXT                       MAPPED_SQL_ID MAPPED_SQL_TEXT
------------- ------------------------------ ------------- ------------------------------
d2vqx2b16ups7 select 22 from dual            355cqk9my0sna select 33 from dual

9fusd37prv595 select 2 from dual             d2vqx2b16ups7 select 22 from dual
```

Note: V$SESSION: SQL_TRANSLATION_PROFILE_ID

Digression: Materialized View Rewrite

There is an API to the Materialized View Rewrite mechanism
Available since 10g?
Direct access provided by dbms_advanced_rewrite
More restrictions

DBMS_ADVANCED_REWRITE restrictions

```
SYS@gluent> @create_rewrite
Enter value for rewrite_name: KSO7
Enter value for from_statement: select max(col1) from kso.skew
Enter value for to_statement: select distinct col4 from kso.skew

BEGIN sys.dbms_advanced_rewrite.declare_rewrite_equivalence (  name
=> 'KSO7',  source_stmt => 'select max(col1) from kso.skew',
destination_stmt => 'select distinct col4 from kso.skew',  validate
=> FALSE,  rewrite_mode => 'GENERAL'  ); END;

*
ERROR at line 1:
ORA-30389: the source statement is not compatible with the
destination statement
ORA-01790: expression must have same datatype as corresponding
expression
ORA-06512: at "SYS.DBMS_ADVANCED_REWRITE", line 29
ORA-06512: at "SYS.DBMS_ADVANCED_REWRITE", line 185
ORA-06512: at line 1
```

DBMS_ADVANCED_REWRITE restrictions

```
SYS@gluent> !oerr ora 30389
30389, 00000, "the source statement is not compatible with the destination statement"
// *Cause:  The SELECT clause of the source statement is not compatible with
//          the SELECT clause of the destination statement
// *Action: Verify both SELECT clauses are compatible with each other such as
//          numbers of SELECT list items are the same and the datatype for
//          each SELECT list item is compatible
```

Demo

If we have internet access ☺

Questions?

Email: kerry.osborne@enkitec.com
Blog: kerryosborne.oracle-guy.com
Twitter: @KerryOracleGuy