Subject: Re: LIST2 Pattern Problem
Posted by ajwid01 on Fri, 19 Jun 2009 13:58:18 GMT
View Forum Message <> Reply to Message

I'll park the primary key problem for right now, until I can properly debug it.

I did have the query pasted in notepad for you, until a scheduled update rebooted my PC (when apparently it shouldn't have), and I lost it.

Will be easy enough to recreate. I'll put back the original query, let it error and then post you the results.

Roughly, from what I recall it's:

I think the HAVING clause (count(dict_table.table_id {?}) gets converted to a WHERE clause, but when it gets re-written by Oracle in the separate oracle php class, it's no longer a COUNT(), as the outer SELECT has converted it to a designated column id from the inner select's alias.

[Hope that makes sense.]

And I think Oracle is still griping about the inner select's in-line query not being in the 'group by'.

[This is the bit that I rewrote as a function call, to avoid the in-line query and so allow a group by on it.]

This is my modified section from the db_table.class.inc:

```
if ($item == 'table_id') {
        // get data from the database, ignoring any tables which have no entries on
DICT_COLUMN
        $this->sql_select     = 'dict_table.table_id, table_desc, count(dict_column.table_id) as
column_count';
        $this->sql_orderby    = 'dict_table.table_id';
        $this->sql_ordery_seq = 'asc';
        $this->sql_from       = 'dict_table '
                        . 'LEFT JOIN dict_column ON
(dict_table.database_id=dict_column.database_id AND dict_table.table_id=dict_column.table_id)
';
        $this->sql_having     = 'column_count > 0';
        $this->sql_groupby    = 'dict_table.table_id,
table_desc,AJW_RAD_GET_REL_TABLE_COUNT(DICT_TABLE.DATABASE_ID,
DICT_TABLE.TABLE_ID) ';
        $data = $this->getData($where);

        // convert each row into 'id=id' in the output array
        foreach ($data as $row => $rowdata) {
           $rowvalues = array_values($rowdata);
           $array[$rowvalues[0]] = $rowvalues[0];
```

```
        } // foreach
        return $array;
    } // if
```

The main bits are the function call for REL_COUNT, and the changing of the HAVING clause away from the count. I think yours USED to be like this, before avoiding the aliased column that Oracle didn't like.

But now, by the time it gets converted from HAVING to WHERE to the Oracle specific PHP class, it can reference the column directly again.

The simple (rough and ready) code for the REL_COUNT replacement was just:

```
CREATE OR REPLACE FUNCTION AJW_RAD_GET_REL_TABLE_COUNT(IN_DB
VARCHAR2, IN_TAB VARCHAR2) RETURN NUMBER
IS

V_RETURN NUMBER;

BEGIN

BEGIN

SELECT COUNT(*)
INTO V_RETURN
FROM DICT_RELATIONSHIP
WHERE
 DATABASE_ID_SNR = IN_DB AND
 TABLE_ID_SNR = IN_TAB;

EXCEPTION
WHEN OTHERS
THEN
 RETURN 0;

END;

RETURN V_RETURN;

END;
/
```

I think that kinda works.

Don't think I changed anything else.