ABS Tech Note: DAL10 DAL and Rdb (6/92)

Revised: 9/2/93
Security: Everyone

ABS Tech Note: DAL10 DAL and Rdb (6/92)

==========

Article Created: 30 June 1992

TOPIC ----------

This technical note discusses restrictions and limitations with DAL and Rdb.

DISCUSSION ----------

The Rdb DBMS adapter for the DAL Server is best viewed as a VAX/VMS C application which uses Rdb's Dynamic SQL interface to access Rdb databases. Accordingly, any DAL user who intends to access Rdb databases should ensure that his or her account is set up with the proper VMS privileges and quotas required by Rdb. DAL users should also note that any current restrictions or problems in Digital's Rdb and Dynamic SQL will also affect the DAL Rdb DBMS adapter.

For example, when Rdb performs arithmetic operations on floating point data types such as the SQL aggregate function AVG(), Rdb rounds any decimal result to a whole number. Consequently, DAL client queries containing aggregate functions on decimal or float datatypes will return rounded numbers when executed in Rdb.

Problem #1

Referring to multiple databases by dbaliasname (AuthID)

To support multiple Rdb databases, the DAL Server Rdb Database Adapter uses the DECLARE SCHEMA statement in Rdb's Dynamic SQL for each database opened. In Rdb, each database or schema is referenced by an "authorization identifier" or AuthID. DAL uses the dbaliasname specified in an OPEN DATABASE statement as the AuthID, or when a dbaliasname is not specified, the database name is used as the alias and AuthID by default. For example:

```
                                                dbaliasname    AuthID
  open database "sample";                          sample      sample
  open database "reservations" alias "test";        test        test
```

Normally, the DAL Server keeps track of the AuthID for you. However, if you mix normal DAL statements with Rdb statements (via the DAL EXECUTE IMMEDIATE statement), you may need to use the AuthID when referring to tables, views, and indexes or when using the Rdb SET TRANSACTION statement. When referring to tables by the dbaliasname in DAL, the correct syntax is:

```
               dbaliasname!table_name        sample!offices
```

Rdb uses a period (.) instead of an exclamation point "!" when referencing a table by the schema's AuthID:

```
               AuthID.table_name             sample.offices
```

(Note: DAL uses the period (.) to prepend an owner name to a table. For example, sample!mark.offices in DAL indicates the table offices owned by mark in the database whose dbaliasname is "sample".)

So, when using DAL's EXECUTE IMMEDIATE statement to set a transaction mode for a database or to reference a table, be sure to use the database's dbaliasname as the AuthID.

For example:

```
  open rdb dbms;
  open rdb database "daldemo" alias  test  in location "msad$system";
  execute immediate "create table small (text char(20))";
  commit;
  insert into test!small values ("xyz");
  execute immediate "update small set text = 'small fails'
          where text = 'xyz' ";
  rollback;
  insert into test!small values ("xyz");
  execute immediate "update test.small set text = 'test.small works'
          where text =   'xyz' ";
  commit;
```

Problem #2

Attempting to query directly against the Rdb System Relations occasionally fails. For example, the following query fails:

```
  SELECT RDB$RELATION_NAME FROM RDB$RELATIONS;
```

Explanation: When DAL is asked to do a query, it checks the syntax and content of the query statement to ensure that the columns and tables asked for exist in the database and to determine the data types of the results. Certain system relations contain columns with a "SEGMENTED STRING" data

type (seg_str). This data type is not supported in SQL interface for Rdb 3.xxx (but will be supported in Rdb 4.0). Even though the column queried is not of the type seg_str, DAL looks at the entire structure (column names, data types, and lengths) to check for correct syntax and content. In doing so, Rdb generates the following error code when it encounters a column with a seg_str: SQL-F-INVSSCONV, Invalid conversion for segmented string column RDB$VIEW_BLR.

WORKAROUND:
Use the execute statement to create a view that specifies the non-seg_str columns desired. Then use DAL to query the view. This way, the DAL server's table look-up won't stumble on columns with the data type seg_str. The following example does this:

```
  Execute immediate "create view temp as
          select rdb$relation_name from rdb$relations";
  commit ;                                    /* don't forget to commit */
  Select rdb$relation_name from temp;
```


Tech Info Library Article Number:11637