DB2 UDB for iSeries Long and Short Identifiers

Kent Milligan

August 27, 2001

This article provides information on customizing your long and short identifiers for your implementation of DB2 UDB. Using these tips helps an application integrate better with existing iSeries commands, utilities, and applications.

Introduction

The iSeries supports two types of SQL identifiers: ordinary identifiers and delimited identifiers. Ordinary identifiers begin with an upper-case letter and are converted to all upper-case letters. A delimited identifier is a sequence of one or more characters enclosed within SQL escape characters. The SQL escape character is a quotation mark, so a delimited column name might be designated as "CompanyName". The maximum length of a delimited identifier includes the two SQL escape characters for column names, but is not included for other SQL names. DB2 UDB for iSeries does not support double-byte table names.

In general, the DB2 UDB for iSeries SQL identifier length maximums are not an issue when porting other databases to the iSeries. DB2 UDB for iSeries column names can be up to 30 characters (really 28 characters if using delimited column names) and most SQL objects such as tables, indexes, procedures, and constraints support a maximum length of 128 characters. Schema and authorization names are the identifier types with the lowest maximum length at 10 characters. All of the SQL limits are documented in the DB2 UDB for iSeries SQL Reference. While DB2 UDB for iSeries supports long SQL identifiers, the commands and interfaces native to the iSeries operating system, OS/400, have a hard maximum of 10 characters for object names. For example, the Save commands that would be used to backup database objects such as tables and indexes only support a 10 character identifier. So how do you backup an SQL object that has an identifier longer than 10 characters? When an SQL identifier longer than 10 characters is specified, DB2 UDB for iSeries will automatically generate a "short" identifier of 10 characters in length that can be used with OS/400 commands and native interfaces. This "short" name is generated for all SQL object names, including column identifiers. These shorter SQL identifiers with the 10 character maximum are also known as system names. The iSeries support for long and short SQL identifiers is very analogous to the file name restrictions when trying to use Windows and DOS utilities on the same object.
The system-generated short names do have a downside since they are generated by the system. First, they are NOT user-friendly. DB2 UDB takes the first five characters of the SQL identifier and then appends a five digit unique number to those five characters. For instance, CUSTOMER_MASTER would have a short name of CUSTO00001. Second and more importantly, these short names are not guaranteed to be consistent across systems or repeated creations of the same SQL object due to the dependencies on creation order and other identifiers that share the same first five characters. Thus, DB2 UDB for iSeries provide SQL syntax to allow you to control the short name.

The RENAME TABLE SQL statement was added to DB2 UDB for iSeries to enable the developer to define the short system name that is assigned to their SQL table. The RENAME TABLE statement can also be used to control the short name for Index and View objects.

In addition, the FOR COLUMN clause was added to the iSeries CREATE TABLE and ALTER TABLE statements to assign a short column name. A short and long column name are needed on some database definitions since there are iSeries programming languages (eg, RPG) that are not able to access table columns with an identifier longer than ten characters. Here is an example of using the FOR COLUMN clause on a CREATE TABLE statement to assign your own short system names for the column definitions:

```
CREATE TABLE dbtest/customer_master
  (customer_name FOR COLUMN cusnam CHAR (20),
   customer_city FOR COLUMN cuscty CHAR(40))
```

With this definition, SQL-based programming interfaces such as ODBC and JDBC would reference the long SQL names (customer_name & customer_city) and legacy application programs would reference the same columns by using the short identifiers (cuscty and cusnam).

Continuing with this example, let's see an example of using the RENAME TABLE statement to control the short table name. To overwrite the system-generated name for customer_master (ie, CUSTO0001), the RENAME TABLE statement will instead assign a short name of cusmst.

```
RENAME TABLE dbtest/customer_master
  TO SYSTEM NAME cusmst
```

After execution of this statement, the SQL table name is `customer_master` and the system table name is `cusmst`. Your SQL requests would reference customer_master and OS/400 system commands such as SAVOBJ (Save Object) or DSPFD (Display File Description) would utilize the short name, cusmst.

If the SQL object happened to be created with a short name as in the following example, then the RENAME TABLE statement can also be used as follows to assign a "long" SQL identifier to that object. After this RENAME request is completed, the result is the same as the previous example — the SQL table name is customer_master and the system table name is cusmst.
CREATE TABLE dbtest/cusmst
   (customer_name FOR COLUMN cusnam CHAR (20),
   customer_city FOR COLUMN cuscty CHAR(40))
RENAME TABLE dbtest/cusmst TO customer_master
   FOR SYSTEM NAME cusmst

Both the short and long names are stored in the DB2 catalogs. If you were not sure of the system
name for a table, then the following query could be run to extract that information from the
SYSTABLES catalog. Similar queries could be executed against the catalog views for the other
types of SQL identifiers.

```
SELECT system_table_name FROM qsys2.systables
   WHERE table_name='CUSTOMER_MASTER' AND table_schema='DBTEST'
```

The SPECIFIC clause can be used to assign a short name when creating Stored Procedures and
User-Defined Functions. When an SQL procedure or function is created, DB2 UDB for iSeries
generates a C program object to implement the business logic. Even though a procedure or
function name has a maximum length of 128 characters, the C program object is limited to 10
characters just like the table object. When you have a procedure or function name that is longer
than 10 characters, the SPECIFIC clause can be used as follows to assign a short name to the C
program object. Instead of having a C program object named UPDAT00001, the program object
will be given UPDSAL as the name.

```
CREATE PROCEDURE UPDATE_SALARY
   (IN EMPLOYEE_NUMBER CHAR (10), IN RATE DECIMAL (6,2))
LANGUAGE SQL
SPECIFIC UPDSAL
MODIFIES SQL DATA
BEGIN
   UPDATE EMP SET SALARY =SALARY +RATE
   WHERE EMPNO =EMPLOYEE_NUMBER;
END
```

With this information on DB2 UDB for iSeries customizations, you are now equipped to tailor
your SQL database definitions to integrate better with existing iSeries commands, utilities, and
applications.

© Copyright IBM Corporation 2001
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)