

Oracle passwords : a few not too well known facts

Although its use is officially undocumented, the keyword `VALUES` followed by a quoted string as in :

```
alter user scott identified by values 'ROAR';
```

is, through technical tips and various publications, well-known by many an experienced DBA to bypass the encryption routines and store - or restore! - the encrypted password 'as is' - This is the technique used by the `become.sql` script freely available at the Oriole web-site (www.oriola.com).

There is, however, more to say about passwords, and quite interesting things indeed. Syntactically speaking, passwords appear, in the normal, documented `alter user` statement not as a quoted string, but as, for lack of a better word, 'text', just as the user's name. Moreover, a

```
desc DBA_USERS
```

will show that the password is defined, as the user name, as `VARCHAR2(30)`. In plain language, it means that passwords are identifiers, exactly like user names, table names, column names... and here comes the famous rule of which most people know only the first part :

- Oracle identifiers are case insensitive, start with a letter and are any combination, up to 30 character long, of letters, digits, and the three special characters `_`, `$` and `#`,
- or Oracle identifiers can be absolutely anything, case sensitive, up to 30 character long when specified between double quotes.

In other words, once specified between double quotes, passwords can contain anything, including control characters. This has some very useful applications, especially where security is a concern.

Most Oracle sites have batch programs running regularly requiring a connection to the database; the ideal way to deal with such a situation is to use externally identified Oracle accounts, but this is not always possible (especially when `SQL*Net` comes into the picture). More often than not, usernames and passwords are coded, sometimes as environment variables; if security is properly ensured, the right to read the files where these values are coded is strictly limited to the file owner. However, this usually leads to a situation where, as modifying a password implies modifying a few other things with the risk of forgetting one, hard-coded passwords are here to stay. It is therefore highly important to choose improbable values for them (the very same thing can be said about Oracle accounts used for database links, as anonymous database links, which forces to change passwords everywhere synchronously are not really practical), and here weird characters can be quite helpful. For instance, inserting a carriage-return character in a password will make using this account interactively rather difficult (although not impossible : with `SQL*Plus`, you have to create a file which contains username/password and run

```
sqlplus @connect_file
```

but you have to know how the password was set-up) .

Creating a user in the following way :

```
create user remote_access identified by "REMOTE ^H_ACCESS";
```

will allow you to create, in the same way, a fairly safe database link which will even have the advantage of viciously looking perfectly dumb on most terminals to anybody querying table `sys.link$` (as passwords no longer appear in `dba_db_links...`)