

ociuldr -- Free text unload utility from dbatools.net

Text file is very useful for data exchange between different database systems. My first text unload program was wrote in Java with JDBC driver, to unload oracle rows as text file, and them load them into Sybase IQ server for analyze. Text file is also very useful for data exchange between different character set oracle databases. While Oracle does not provide such a tool for us, anybody know why?

Tom Kyte published two script on AskTom, first script is SQL*Plus based, second script is wrote in Pro*C. Currently ociuldr is not well known by people. Why did I write this utility? The first script is not so flexible, and the second script is not powerful enough and need to be recompiled under different oracle client version. Let's take a look at the features ociuldr provides for you:

- OCI based, as faster as Tom Kyte's Pro*C utility
- Run smoothly with recompile under Oracle client 8i/9i/10g
- Specify any character or string as field separator and row separator
- Generate sqlldr control file for loading the data to another database
- More performance tuning related option
- Write message to log file
- Exit or return code for integrate into other script

Let's start to learn how to use this free utility.

Syntax and command line option

Ociuldr is a command line tool, not GUI based, you may not like command line tool. But for me, it enable write once and compile it on multiple different platform. I wrote it on windows, and it was successfully compile under Linux/Unix. I will provide the binary file of Windows, RedHat Linux and Solaris Sparc64 for you. This script was also successfully compiled on AIX platform. I cannot provide the binary files on all the platforms for you, so I put the source code on my personal web site for free download, if you cannot find the platform binary file, you can compile your own version:

- Binary: <http://www.dbatools.net/software/ociuldr.zip>
- Source: <http://www.dbatools.net/software/ociuldr.c>

Run ociuldr without any command line parameter to get the help information:

```
C:\MYDUL>ociuldr
Usage: ociuldr user=... query=... field=... record=... file=...
(@) Copyright Lou Fangxin 2004/2005, all rights reserved.
Notes:
    -si    = enable logon as SYSDBA
    user   = username/password@tnsname
    sql    = SQL file name
    query  = select statement
    field  = seperator string between fields
    record= seperator string between records
    file   = output file name(default: uldrdata.txt)
    read   = set DB_FILE_MULTIBLOCK_READ_COUNT at session level
    sort   = set SORT_AREA_SIZE at session level (UNIT:MB)
    hash   = set HASH_AREA_SIZE at session level (UNIT:MB)
    serial= set _serial_direct_read to TRUE at session level
    trace  = set event 10046 to given level at session level
    table  = table name in the sqlldr control file
    log    = log file name, prefix with +to append mode

    for field and record, you can use '0x' to specify hex character
code,\r=0x0d \n=0x0a |=0x7c ,=0x2c \t=0x09
```

Let's take a look at each of these command option:

- **-si** and **user=**

-si tell ociuldr to login to database as sys user (sysdba role), user tell ociuldr the connection information by "username/password@tnsname" format. You must provide one of these two option. And you must give the password in the command line when you specify the user option, ociuldr will not info you to input the password. On oracle 10g, you may not be able to use "-si" option.

- **sql=** and **query=**

This two option will tell ociuldr the SQL query to run to get the rows. Use query option if you want to specify a simple query in command line. If you have a complex query, you should create a SQL file, and the specify sql option as the SQL file name to read the SQL query. Don't append ":" or "/" to the end of the SQL query, and you can only write one SQL in the SQL file, multiple SQL queries are not allowed.

- **field=** and **record=**

We need specify a character or a string as separator between the fields/columns, and between different rows. Field option used for fields/columns separator, the default value is “|”. Record used for row separators, the default value is newline (windows style, \r\n). When you choose the separators, please use characters or strings which will not appear in the data values, you can use any character even invisible characters as separators by specifying the ASCII code, please visit the page http://www.anysql.net/developer/ascii_hex_table.html for the ASCII code of each invisible character. As a matter fact, I always use 0x07 as the fields/columns separators, and use 0x06 for row separators, the two characters are not be able to be inputted by keyboard.

Few people are asking how to generate Excel CSV format file, actually it's a text file with fields/columns separator “,” and row separator newline.

- **file=**

This option tells ociuldr the output data file name, default value is “uldrdata.txt” in the directory where you run the ociuldr. If batch option is provided to generate multiple output data file, please use “%d” for the file sequence, for example “text_%d.txt”.

- **read=**

In Oracle, DB_FILE_MULTIBLOCK_READ_COUNT may be used for performance tuning. If this option is set to a number value, oracle will set this parameter at session level. For more information of this parameter, check document of oracle tuning.

- **sort=**

In Oracle, SORT_AREA_SIZE and SORT_AREA_RETAINED_SIZE may have big performance impact to SQL query with sort operation. If this option is set, ociuldr will change these two parameters at session level before run the SQL query, the unit is MB. For more information of these two parameters, check document of oracle tuning.

- **hash=**

In Oracle, HASH_AREA_SIZE may have performance impact with SQL query use hash join method between multiple tables. If this option is set, ociuldr will change

this parameter at session level before run the SQL query, the unit is MB. For more information of this parameter, check document of oracle tuning.

- **serial=**

In Oracle, `_serial_direct_read` parameter can be set at session level with TRUE or FALSE value, if it's set to TRUE, oracle will use direct path read access method for table access without index available. If this option is set to non zero value, `ociuldr` will set this parameter to TRUE at session level, it may be helpful for performance.

- **trace=**

In Oracle, event 10046 can be used to trace the execution of a SQL query. You can set this option to different level (valid options are 1,4,8,12) for trace of the SQL query. By default `ociuldr` will not set this event.

- **table=**

If this option is set, `ociuldr` will generate a SQL*Loader control file. This option's value will be the target table name of the SQL*Loader control file. By default this option's value is NULL, so `ociuldr` will not generate the control file for you unless this option is set. The control file name will be the option value with a suffix of `"_sqlldr.ctl"`.

- **head=**

Sometime we hope to print the column name as the first line of the output data file, by default this option's value is off, it will not print the column name. If you set it to yes or on, `ociuldr` will print the column name as the first line separated by fields separator. So when you reload the data file, you have to skip the first line.

- **batch**

When you want to unload a lot of rows, you may want `ociuldr` to write them to multiple data files to avoid one extremely huge file. By the batch option (any number larger than 1), it will tell `ociuldr` to shift to new output file after given batches (each batch is 500000 rows). If this option is specified, keep attention to the file option, it should contains `"%d"` pattern. For example:

TEST_%d.TXT

The file generated will be: TEST_1.TXT, TEST_2.TXT,

- **log=**

When unloading, ociuldr will write out message to tell you how many lines have been unloaded, and write the error message when anything goes wrong, by default this message will be write to screen, the standard output device. When you integrate ociuldr into your script, you may want ociuldr to write these message to a log file for later trouble shooting. This option will tell ociuldr a log file name to write message to. There are two mode, overwrite mode (default) or append mode (by put a plus as the beginning of the log file name).

- **long=**

We may use LONG type to store some characters longer than 4000 bytes, for example, the complex SQL definition of a report, a detailed description of a product. This option set the maximum length ociuldr will read from the database. The default value is 4000, this value is set for client memory saving.

- **array=**

I add this option to dynamically set the array fetch size, the minimum value is 5 and the maximum value is 2000. When the table have LONG column, the long option value multiple array size cannot exceed 100MB. It means when you unload data from long column, you should decrease your array size, else this tool can consume a lot of memory.

- **mode=**

There are four options for sqlldr data loading, the default mode is INSERT. By this value, you can only load data into empty table, else you must edit the generated control file. With this option, you can change it to “APPEND” or “REPLACE” or “TRUNCATE”. The “APPEND” mode will load rows to tables with existing data.

Take these command line into account, you can see that ociuldr is more powerful and more flexible, this script is not a simple copy of the unload.pc from AskTom. It's wrote by OCI, fully rewrite by me.

Run SQL “Select * from tab”

Now we will start a simple case, we will choose sharp (“#”) as fields/columns separator, and use the default value (newline) as rows separator. Make sure you have

install the Oracle client on the machine on which you run ociuldr utility, and also setup the Oracle SQL*Net for remote connection.

The first example will run on Windows, connect to user: anysql (password: anysql) of the test database (connect string : test) :

```
C:\MYDUL>ociuldr user=anysql/anysql@test query="select * from tab"
field=#

      0 rows exported at 2007-03-06 11:40:33
     39 rows exported at 2007-03-06 11:40:33
      output file uldrdata.txt closed at 39 rows.
```

Let's check the content of the output file (default file name : uldrdata.txt):

```
C:\MYDUL>cat uldrdata.txt
A#TABLE#
A_V#VIEW#
CCC#TABLE#
FACT_SALES#TABLE#
MV_FACT_SACLES#TABLE#
OBJD_LIST#TABLE#
STAT_T_HASH#TABLE#
STAT_T_HASH_P2#TABLE#
STAT_T_HASH_P3#TABLE#
STAT_T_PARTDEMO#TABLE#
.....
```

Run SQL “Select * from tab” again

The second case is run the same SQL again, but with more command line options:

- Fields/columns separator is 0x07
- Rows separator is 0x06
- Output file name is anysql.txt
- Generate SQL*Loader control file, target table name is “ANYSQL_TAB”
- Put the SQL query in a SQL file named “tab.sql”

Prepare the SQL text file tab.sql :

```
C:\MYDUL>cat tab.sql
select *
from
tab
```

Run ociuldr to unload the rows :

```
C:\MYDUL>ociuldr user=anysql/anysql@test sql=tab.sql field=0x07
record=0x06 file=anysql.txt table=anysql_tab

    0 rows exported at 2007-03-06 11:54:49
    39 rows exported at 2007-03-06 11:54:49
    output file anysql.txt closed at 39 rows.
```

Check the control file of SQL*Loader named “anysql_tab_sqlldr.ctl” :

```
C:\MYDUL>cat anysql_tab_sqlldr.ctl
--
-- Generated by OCIULDR
--
OPTIONS(BINDSIZE=4194304,READSIZE=4194304,ERRORS=-1,ROWS=50000)
LOAD DATA
INFILE 'anysql.txt' "STR X'06'"
INTO TABLE anysql_tab
FIELDS TERMINATED BY X'07' TRAILING NULLCOLS
(
  TNAME CHAR(30),
  TABTYPE CHAR(7),
  CLUSTERID CHAR(40)
)
```

Create a new table named “anysql_tab” with the structure of query result :

```
SQL> desc anysql_tab
Name                               Null?      Type
-----
TNAME                               NOT NULL   VARCHAR2(30)
TABTYPE                             VARCHAR2(7)
CLUSTERID                            NUMBER
```

Load the rows into new table with the control file by SQL*Loader :

```
C:\MYDUL>sqlldr anysql/anysql@test control=anysql_tab_sqlldr.ctl

SQL*Loader: Release 10.2.0.1.0 - Production on Tue Mar 6 12:01:21
2007

Copyright (c) 1982, 2005, Oracle. All rights reserved.

Commit point reached - logical record count 39
```

Of cause you need to check the rows load, and check whether there is bad file or discard file generated by SQL*Loader utility.

Test the log option

```
ociuldr user=ansyql/anysql@s8i query="select * from tab" log=tab.log
ociuldr user=ansyql/anysql@s8i query="select * from tab" log=+tab.log
```

The exit code or return code

Exit code or return code is very useful when you embed ociuldr into your script, and you want to know whether the ociuldr run successful or fail with error. The return code of ociuldr are :

```
0 = Successful
1 = Cannot login to database
2 = Cannot create cursor handle
3 = Cannot prepare SQL statement
4 = Cannot execute SQL query
5 = Cannot get the metadata of the result set
6 = Cannot create output file
7 = Oracle error found when fetching rows, such as ORA-01555 etc.
```

Let's test different case. When we cannot connect to the database :

```
bash-2.03$ ./ociuldr.bin user=anysql/anysql@test query="select *
from tab"
Cannot connect as anysql/anysql@test.
Connection failed. Exiting...
bash-2.03$ echo $?
1
```

When wrong SQL are provided :

```
bash-2.03$ ./ociuldr.bin user=anysql/anysql@test query="select *
from tabl"
ORA-00942: table or view does not exist
bash-2.03$ echo $?
3
```

When no permission of creating the output file :

```
bash-2.03$ ./ociuldr.bin user=anysql/anysql@test query="select *
from tab" file=/test.txt
    15500 bytes allocated for column TNAME (1)
     4000 bytes allocated for column TABTYPE (2)
    20500 bytes allocated for column CLUSTERID (3)

ERROR -- Cannot write to file : /test.txt
bash-2.03$ echo $?
6
```

How to run under Unix/Linux OS?

In 64 bits Oracle version, we can choose to use 64 bit client version or 32 bits client version. Before 9i, the 32 bits client library is put under \$ORACLE_HOME/lib directory, and 64 bits client library is put under \$ORACLE_HOME/lib64 directory. Since 9i version, 32 bits is put under \$ORACLE_HOME/lib32 directory, 64 bits client is put under \$ORACLE_HOME/lib directory. To make the ociuldr run well for you, I want to

tell you that the binary on my site was compiled with 32 bits oracle client, so you must set the environment variable LD_LIBRARY_PATH or SHLIB (under AIX) to point to the 32 bits client library directory.

When I run ociuldr, I rename the binary file to “ociuldr.bin”, and then create a shell script named “ociuldr” as following :

```
#!/bin/sh

NLS_DATE_FORMAT="YYYY-MM-DD HH24:MI:SS"

if [ "A${ORACLE_HOME}A" = "AA" ]; then
    echo "ORACLE_HOME environment variable not setted."
    exit
fi

if [ "A${LD_LIBRARY_PATH}A" = "AA" ];then
    LD_LIBRARY_PATH=/lib:/usr/lib
fi

if [ -d ${ORACLE_HOME}/lib32 ]; then
    LD_LIBRARY_PATH=${ORACLE_HOME}/lib32:${ORACLE_HOME}/lib:${LD_LIBRARY_PATH}
else
    LD_LIBRARY_PATH=${ORACLE_HOME}/lib:${ORACLE_HOME}/lib64:${LD_LIBRARY_PATH}
fi

export LD_LIBRARY_PATH NLS_DATE_FORMAT
ociuldr.bin "$1" "$2" "$3" "$4" "$5" "$6" "$7" "$8" "$8"
```

This shell script will check where ORACLE_HOME is set, and then set the correct LD_LIBRARY_PATH to start the utility.

Compile your binary under Unix/Linux

If you cannot find the binary of your platform, download the source code and compile it by yourself.

The source code lines of ociuldr are all in one file named ociuldr.c. You need to install the C compiler first. Following is the syntax to compile the binary with gcc on common unix platforms :

```
gcc -m32 -g -Bsymbolic -t -D_LARGEFILE64_SOURCE -
D_FILE_OFFSET_BITS=64 -I${ORACLE_HOME}/rdbms/demo -
I${ORACLE_HOME}/rdbms/public -L${ORACLE_HOME}/lib32 -Wl,-i -o
ociuldr.bin ociuldr.c -lm -lcintsh -Wl,-Bdynamic
```

If you have problem with the compiling, search on the google to find out how to compile the OCI program, or get in touch with me.

Suggestion & Feedbacks

You suggestions or feedbacks are welcome. You can tell me or just modify the source code, but please info me.

Thanks for choose this free utility, always visit <http://www.dbatools.net> for the latest update of the free utility.