

A DBA's good friend, DBCA (ver 0.5)

Aman Sharma

Introduction

Once you are in to some profession, there are some basic duties which you have to perform related to that profession. Things which after some time, are not even considered some thing really important. How many times you think that while driving you have applied brakes while cooking something really special, how neatly and fastly you have chopped those vegetables? Certain things become a part of our daily routine which if we look are so important and require very minute details to be taken care of, yet appear so simple!

For a database administrator, creating a database is something which appears to be an obvious and very simple task. Let's face it, what good would be a dba who doesn't even know how to create a database? If he can't create one, how on the earth he is going to manage it? Its such a basic thing isn't it, how one can be wrong about it? And it being so simple, it should be matter of not more than few minutes to create a database you may say, isn't it? How difficult it can be to type a simple command like create database and voila, we have a nice and shiny new database! Life would have been so much better if we could be in an ideal world where all what we could think would have come true but its not. And so are some tasks which appear very easy but hide a lot of complexity underneath them.

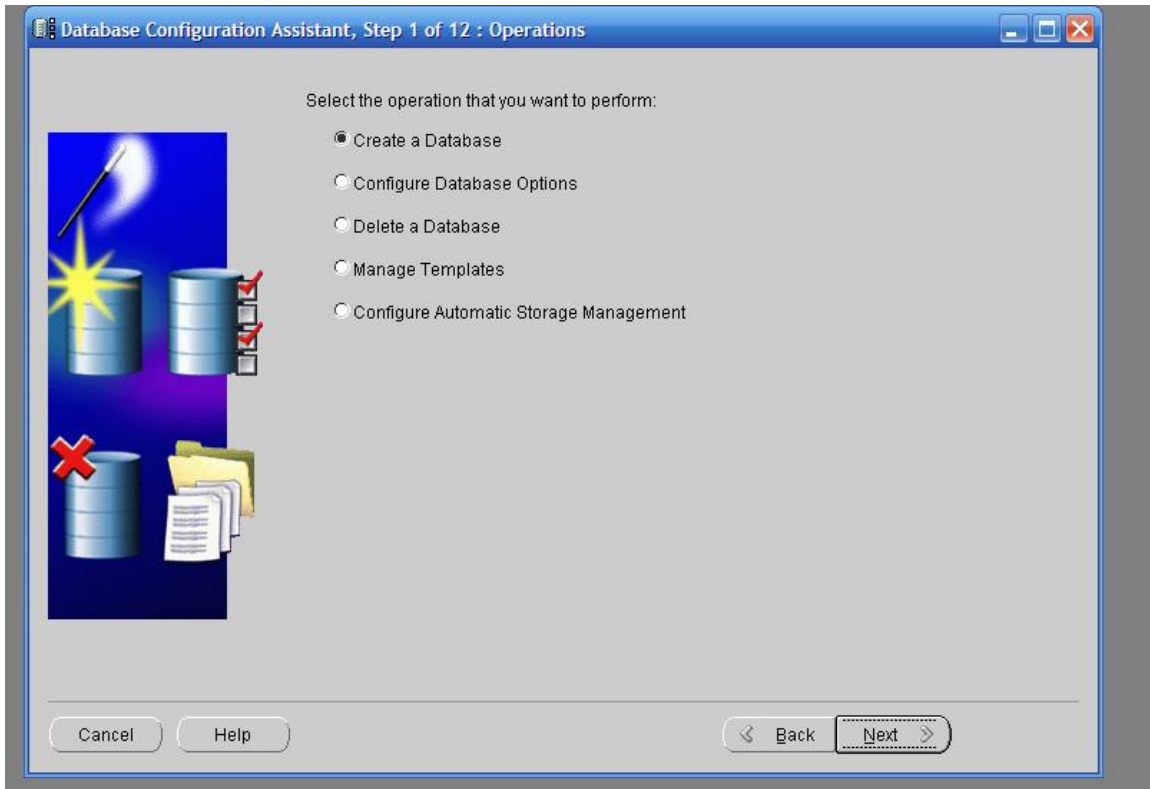
But wherever there is a problem, there is a solution for it some where as well! If writing a command and handling its details can be tough, there can be some other way out to do the same task, using some other things, like a tool which does everything with just a button click, leaving no place for mistakes. We are going to talk about such a tool today which has made lives of many dba's much simpler when it comes to creation of a database, *Database Configuration Assistant aka DBCA*.

Getting Started With DBCA

Database Configuration Assistant (DBCA) is a Java based stand alone tool which comes very handy when we have to create, configure and even drop databases. From 10g Release 2 onwards, this has been optimized to create and manage Automatic Storage Management (ASM) instance as well. You can start DBCA by simply typing "dbca" over your terminal prompt while connected as the user who is the oracle software owner like below

```
$dbca &
```

This would come up with a screen like the following with various options that can be used to manage the database. These options decide the next screens that we would see when we click on the Next button.



Options Presented by DBCA

a) *Create a Database*

This option is among the most popular and useful options. As the name suggests, this is used to create a new database. The next screens present the options of selecting either predefined templates like Transaction Processing, Data Warehouse (we shall discuss templates in a while), choosing the database name or its other properties like SGA/PGA size. This also gives the option to save our choices in the form of scripts that can be used by us over some other host to recreate a database with the similar settings.

b) *Configure Database Options*

This option is used to configure those options which we either missed or didn't choose deliberately while creating the database. Depending on the option that we want to configure, we are presented with the configuration details and tablespaces where those options would reside.

c) *Delete Database*

This should come as the most simplest yet most deadly options. Using this would drop the database and would also remove the underlying files of it. If you are on

Windows environment, this would also remove the database service from the operating system.

d) Manage Templates

This option is used to create new templates and to delete the existing ones. In addition to that, we can create templates with or without data from the predefined database templates as well. While choosing the data also coming along with the template creation, the database files and the schemas within them with their data also is taken. This is done in a compressed format.

e) Configure Automatic Storage Management

From 10g release 2 onwards, DBCA and OUI, both are optimized to create and manage the instance of Automatic Storage Management. This option is used to create and manage the ASM instance if it's not created already. This takes care of creating the Server Parameter file and password files of the ASM instance and initializing it as well.

Getting to know DBCA Little More

DBCA has really made the life very simple with the introduction of the templates and the scripts that it creates when a database is created through it. Using these scripts, one can understand a lot better that how to go ahead and create manually such files plus the same scripts and templates can be used to create another database with similar properties over some where else too. This is true for both single and for RAC database as well as DBCA is usable in both the environments.

Templates of DBCA

There are 4 predefined templates that come with the software itself. All the templates, either predefined or newly created are in the XML format. One such template is presented below which is used to create a new database which was manually customized and didn't use any of the predefined templates.

```
<Database Template name="newdb" description="" version="10.2.0.0.0">
```

```
<Common Attributes>
```

```
<option name="ISEARCH" value="false"/>
```

```
<option name="OMS" value="false"/>
```

```
<option name="JSERVER" value="true"/>
```

```
<option name="SPATIAL" value="false"/>
```

```
<option name="ODM" value="false">
```

```
<tablespace id="SYSAUX"/>
```

```
</option>
```

```
<option name="IMEDIA" value="true"/>
```

```
<option name="XDB_PROTOCOLS" value="true">
```

```
<tablespace id="SYSAUX"/>
```

```
</option>
```

```

<option name="ORACLE_TEXT" value="false">
  <tablespace id="SYSAUX"/>
</option>
<option name="SAMPLE_SCHEMA" value="false"/>
<option name="CWMLITE" value="false">
  <tablespace id="SYSAUX"/>
</option>
<option name="EM_REPOSITORY" value="false">
  <tablespace id="SYSAUX"/>
</option>
<option name="HTMLDB" value="false"/>
<option name="NET_EXTENSIONS" value="false"/>
</CommonAttributes>
<Variables/>
<CustomScripts Execute="false"/>
<InitParamAttributes>
  <InitParams>
    <initParam name="pga_aggregate_target" value="90" unit="MB"/>
    <initParam name="processes" value="150"/>
    <initParam name="db_recovery_file_dest_size" value="2048" unit="MB"/>
    <initParam name="control_files"
value="(&quot;{ORACLE_BASE}\oradata\{DB_UNIQUE_NAME}\control01.ctl&quot;;,
&quot;{ORACLE_BASE}\oradata\{DB_UNIQUE_NAME}\control02.ctl&quot;;,
&quot;{ORACLE_BASE}\oradata\{DB_UNIQUE_NAME}\control03.ctl&quot;);"/>
    <initParam name="sga_target" value="160" unit="MB"/>
    <initParam name="compatible" value="10.2.0.1.0"/>
    <initParam name="background_dump_dest"
value="{ORACLE_BASE}\admin\{DB_UNIQUE_NAME}\bdump"/>
    <initParam name="job_queue_processes" value="10"/>
    <initParam name="db_name" value="newdb"/>
    <initParam name="user_dump_dest"
value="{ORACLE_BASE}\admin\{DB_UNIQUE_NAME}\udump"/>
    <initParam name="dispatchers" value="(PROTOCOL=TCP) (SERVICE={SID}XDB)"/>
    <initParam name="audit_file_dest"
value="{ORACLE_BASE}\admin\{DB_UNIQUE_NAME}\adump"/>
    <initParam name="db_domain" value=""/>
    <initParam name="open_cursors" value="300"/>
    <initParam name="db_block_size" value="8" unit="KB"/>
    <initParam name="db_recovery_file_dest" value="{ORACLE_BASE}\flash_recovery_area"/>
  </InitParams>
</InitParamAttributes>
<output snipped>
</DatabaseTemplate>

```

As can be seen from the above template, all the options that are needed to create a new database are in it. All the templates, either predefined or new are stored in the *Oracle_home/assistants/DBCA/templates*. There are 4 templates stored here in this directory, 3 with **.DBC* extension and one with **.DBT* extension. These are,

- 1) *Data Warehouse*
- 2) *Transaction Processing*
- 3) *General Purpose*
- 4) *New Database*

You would see that out of the above 4, 3 templates are associated with the *.dbc* extension, for example *Transaction_processing.dbc* and one is associated with the *.dbt* template which is just one, *New_database.dbt*. The difference lies among these 4 is whether they support the creation of *Seed database*, a starter database recommended to be created before one attempts to create a real production database. Creating this database would bring out any sort of issues which may get arise while creating an actual one too, helping a dba to troubleshoot those issues with this database only.

Based on the above, the templates can be categorized into two parts as well,

- a) *Seed templates*
- b) *Non-seed templates*

The extensions with a suffix of *.dbc(Database Clone)* are called *Seed Templates* which actually contain a compressed file, storing the information of the mandatory files required to create the database. There is normally a file available with *.DFJ* extension which contains the files in the compressed format. The DBCA when runs, unzips this file and extracts the data files over the device. The database is then reset and the name of the database is changed to the name that one mentions at that time.

The other template is appended with the *.dbt (Database Template)* extension are called *Non Seed Templates*. This doesn't come up with the default datafiles and is used normally to create a new database which is customized by the user. This is the most useful template as here; almost all the options for both database and its files can be altered according to our own choice.

Oracle comes up with some samples schemas which are helpful to test out various sql features with the preloaded data that they have. These schemas can be created with the starter database that is loaded with the Seed templates. If you are willing to create the sample schemas like, HR, OE, SH and so on, Oracle comes up with a default file which contains the commands to create these schemas and populate them with data. There is an export dump file which is extracted and imported in the database to make these schemas. The export file has a default name *Example.dmp*. The file actually is

not a real dump file and is a database backup file created by RMAN. This file is a transportable tablespace file which is linked to a tablespace Example which comes up preloaded as *Example.dfb* (*Datafile backup*). The *example.dmp* is linked to a sql file, *mkplug.sql* which extracts the files to the Example tablespace using package *dbms_backup_restore* into the Seed database. In other words, the Example tablespace is *plugged into* the target database. Once done, the *example.dmp* is loaded into it, finally creating the sample schemas and their data.

Here is output of the extraction of the *example.dmp* file while being imported, Export file created by EXPORT:V10.02.01 via conventional path

About to import transportable tablespace(s) metadata...

import done in WE8MSWIN1252 character set and AL16UTF16 NCHAR character set
export client uses US7ASCII character set (possible charset conversion)

. importing SYS's objects into SYS

```
"BEGIN sys.dbms_plugts.beginImport ('10.2.0.1.0',1,'2000',7,'Microsoft Win"  
"dows IA (32-bit)',51896,43851,1,1,1,0); END;"  
"BEGIN sys.dbms_plugts.checkCompType('COMPATSG','10.2.0.1.0'); END;"  
"BEGIN sys.dbms_plugts.checkUser('IX'); END;"  
"BEGIN sys.dbms_plugts.checkUser('HR'); END;"  
"BEGIN sys.dbms_plugts.checkUser('OE'); END;"  
"BEGIN sys.dbms_plugts.checkUser('PM'); END;"  
"BEGIN sys.dbms_plugts.checkUser('SH'); END;"  
"BEGIN sys.dbms_plugts.beginImpTablespace('EXAMPLE',6,'SYS',1,0,8192,1,552"  
"765,1,2147483645,8,128,8,0,0,0,8,3896376132,1,33,536653,NULL,0,0,NULL,NULL)"  
"; END;"  
"BEGIN sys.dbms_plugts.checkDatafile(NULL,3896376132,5,12800,6,5,4194302,8"  
"0,536661,552765,1,20971522,NULL,NULL,NULL); END;"  
"BEGIN sys.dbms_plugts.commitPluggable; END;"  
"BEGIN sys.dbms_plugts.endImport; END;"
```

Import terminated successfully without warnings.

We can see the default users being a part of the dump file. Note that these schemas must not be used for anything else except for testing and demonstration. Also if you miss creating them while creating the starter database, the scripts needed to create them are *not* given with the default install media from 10g onwards. You need to download and install Companion disk which comes up with a script *mksample.sql* which is the master script to create them.

The sample schemas are renamed in 11g to Example Schemas.

DBCA Modes

DBCA can run in 3 different modes depending on the requirement. These are,

1) *Interactive*

- 2) *Progressive*
- 3) *Silent*

Interactive mode is the most common mode and is self explanatory as well. This mode requires the user who is running DBCA to answer all the questions asked by DBCA. Once the answers to the questions are received, user is prompted to save the answers in the form of either a script file or template. Both also can be created and the database is finally started to get created. This mode can be used even when the database is going to get created afterwards the software is installed.

Progressive mode is the mode which is shown when the database is getting created at the time of the oracle software installation and we have chosen database creation with it. At that time, OUI calls DBCA to create the seed database. In this mode, only the progress bar is visible for the database creation and the customizations of it are not allowed.

Running DBCA in *Silent* mode

Silent mode is the mode that is the fastest mode and works like a charm if all the options needed by it are given correctly. There are no prompts or progress bar shown and the entire information is stored in the log files which get created with the installation. This mode is started from the command prompt and depends on several conditions that can be used while calling the binary. For example, we can run DBCA in silent mode with the seed template or with a non-seed template. This can be made more complex using templates either from an existing database with the data or without the data. You can see all the possible options which can be used while running DBCA in the silent mode using

```
$dbca -help
```

This would enlist all the options that can be used.

Let's try to run DBCA in silent mode using the non-seed template of new database. We shall be using the non-seed template of New Database. The command used and its snipped output is shown below,

```
E:\>dbca -silent -createDatabase -gdbname test -
templateName
E:\oracle\product\10.2.0\db_1\assistants\dbca\templates\New
_database.dbt
Creating and starting Oracle instance
1% complete
4% complete
Creating database files
8% complete
Creating data dictionary views
9% complete.....
20% complete
21% complete
```

Adding Oracle JVM
22% complete.....
42% complete
Adding Oracle Data Mining
83% complete
Adding Enterprise Manager Repository
84% complete
86% complete
88% complete
Completing Database Creation
99% complete
100% complete

The log of the creation of the database would be at *Oracle_Home/cfgtoollogs/dbca/sid/sid.log* so for our Test database, it would be *TEST.log*. Here is the partial output of the log file of the Test database creation,

Creating and starting Oracle instance

DBCA_PROGRESS : 1%

DBCA_PROGRESS : 4%

Creating database files

DBCA_PROGRESS : 8%

Creating data dictionary views

DBCA_PROGRESS : 9%

DBCA_PROGRESS : 10%

DBCA_PROGRESS : 12%

DBCA_PROGRESS : 13%

DBCA_PROGRESS : 14%

DBCA_PROGRESS : 15%

DBCA_PROGRESS : 17%

DBCA_PROGRESS : 18%

DBCA_PROGRESS : 20%

DBCA_PROGRESS : 21%

Adding Oracle JVM

DBCA_PROGRESS : 22%

DBCA_PROGRESS : 28%

DBCA_PROGRESS : 34%

DBCA_PROGRESS : 40%

Besides this log, the folder would also contain all the creation scripts used by the tool to create the database and to add any components to it. These scripts can be used by the DBA to monitor what really happened in the backend and also in the case of any issues, can serve as the first place to look for troubleshooting. When DBCA finishes the creation of the database, couple of scripts get created under *\$OH/cfgtoollogs/dbca/<sid>*. These scripts contain all the logs of the events that happened within the database creation. Among all the scripts, there is a script with the name, *postDbcreation.log*. This should contain the final message with the database shown as opened, meaning that the creation was successful.

The default location of the data files created by dbca in silent mode will be under `$ORACLE_BASE/oradata`. For example, if you would create a database with the name TEST on a windows machine with Oracle installed on its E drive, the location of the datafiles by default would be under, `E:\oracle\product\10.2.0\oradata\TEST`. If you don't want this as the location of storage for the datafiles of your database, you can use the switch `-datafileDestination` which will set a one stop destination for all database files under the directory name that you would put here. For example, if you would use `-datafileDestination= E:\testdatabase` than under Testdatabase folder, all the files for the Test database would get created. This option is much suitable when you are using a file system based destination for your files. If you are going to *use Raw Devices* than you can use `-datafileNames` switch which points to a file that would contain all the files needed for the database like data, redo and control files mapped with the specific raw file name with a key value pair. From 10gR2 onwards, with ASM also available as an option for the storage, that can also be specified while creating the database.

Running DBCA in Debug mode

At times, it does happen that the DBCA appears to be running but there is no progress shown over the progress bar. And waiting for hours, all of a sudden it disappears, leaving the person scratching his head that what really happened behind the scenes. For such kind of issues, fortunately DBCA can be run in the Debug mode which creates a trace of the information at a more detailed level, helpful for troubleshooting. This mode is not enabled by default and for doing this, you need to edit the binary that runs DBCA.

To enable the trace mode, go to `$OH/bin` and find the dbca binary. Open it with your favorite editor and find a line (almost at the end of the file) which starts with `#Run DBCA`. This contains an entry similar to this,

```
$JRE_DIR/bin/jre -DORACLE_HOME=$OH -DJDBC_PROTOCOL=thin -
mx128m -classpath
    $CLASSPATH oracle.sysman.assistants.dbca.Dbca $ARGUMENTS
```

In this add the following,

```
-DTRACING.ENABLED=true -DTRACING.LEVEL=2
```

So the final part should look like,

```
$JRE_DIR/bin/jre -DORACLE_HOME=$OH -DJDBC_PROTOCOL=thin -
mx128m
    -DTRACING.ENABLED=true -DTRACING.LEVEL=2 -classpath
$CLASSPATH
    oracle.sysman.assistants.dbca.Dbca $ARGUMENTS
```

The same is true for Windows environment only with just one difference that there would be a batch file of DBCA under your Oracle Home's BIN folder. There would be an entry like following in the window version of DBCA,

```
"E:\oracle\product\10.2.0\db_1\jdk\jre\BIN\JAVA" -  
DORACLE_HOME="%OH%" -DJDBC_PROTOCOL=thin -mx128m  
oracle.sysman.assistants.dbca.Dbca %*
```

You need to edit it with the tracing switch just like we did it for Linux.

To enable trace in pre 10g databases, you need to get the output collected by yourself in some file using the redirectional symbols like

```
$dbca > trace.log
```

From 10.2 onward, this mode is by default on for the analysis purpose. The detailed trace would be under the \$OH/cfgtoollogs/dbca/trace.log . In 11g, the location is changed to Diagnostic_Dest wherever you set it to be.

Summing Up

DBCA is a great tool for both seasoned and new dba's as it presents both the simple and complex ways to do the same thing. If you want to take control in your hands, silent mode is what you are looking for otherwise; the traditional GUI serves the best for a fresher who finds writing a long Create Database command hectic. With the enhanced support for ASM and introduction of templates, it's really some thing which can not get unnoticed if you are serious about being a DBA.

References

Oracle Documentation
Oracle Technology Forum threads
Metalink