Doc ID: Note:1006497.6
Subject: How to Create a Database in the UNIX Environment
Type: FAQ
Status: PUBLISHED

Content Type: TEXT/X-HTML
Creation Date: 23-DEC-1994
Last Revision Date: 06-AUG-2003

How to Create a Database in the Unix Environment

PURPOSE
This entry includes the following four methods for creating a database within the UNIX environment and Oracle7:

SCOPE & APPLICATION
To help ensure a better understanding of the corresponding created database files, it is recommended that the following documentation is also referenced:

Administrator's Guide, Creating A Database: Chapter 2
SQL Language Reference Manual, Create Database Command

Method I - Creating the Database Manually

Setting the Unix Enviroment

When creating a database, Oracle looks at the environment to see which database to create. Therefore, it is essential that before creating a database your environment is set with the database name. If you are creating a second database, be sure to use a new database name.
The following environment variables need to be set:

ORACLE_SID - set to the database name you wish to create
ORACLE_HOME - set to full pathname of the Oracle system home directory
PATH - needs to include $ORACLE_HOME/bin

To set your Unix environment use the following commands depending on the Unix shell you are using:

```
sh  - ORACLE_SID XXX;export ORACLE_SID
csh - setenv ORACLE_SID XXX
```

Check to be sure it was changed:

```
echo $ORACLE_SID
```

<NEW_NAME>

NOTE: Not changing the ORACLE_SID environment and running the create database may wipe out your existing database and all of its data.

Creating the Database

This method involves typing the create database statement within SQL*DBA or Server Manager. Using this method allows for more flexibility such as specifying the MAXDATAFILES parameter or specifying multiple SYSTEM tablespace

database files. However by doing this manually there is also a greater possibility of syntax errors. In addition there is no logfile automatically created to record the options which have been specified.

Steps for Method I:

1. Set the UNIX environment (SEE SETTING UNIX ENVIRONMENT SECTION ABOVE).

2. Create new init<sid>.ora and config<sid>.ora files for your new database by copying the default ones provided by Oracle:

```
% cp $ORACLE_HOME/dbs/init<sid>.ora $ORACLE_HOME/dbs/initNEW_NAME.ora
% cp $ORACLE_HOME/dbs/config<sid>.ora
$ORACLE_HOME/dbs/configNEW_NAME.ora
```

3. Change db_name parameter in the new initNEW_NAME.ora from DEFAULT to the new database name and specify the control file's name and location (this is often found in the config<sid>.ora file).

   NOTE: If the database name is not specified when using the create database statement, the name specified for the initialization parameter "DB_NAME" in the init<sid>.ora file is used.

4. Startup SQL*DBA or Server Manager in line mode:

```
sqldba lmode=y
```
or
```
svrmgrl
```

   Note: We will use Server Manager in our examples. If you are using SQL*DBA (versions prior to V7.3.x), the commands are the same.

5. Connect to the instance, and startup in a 'NOMOUNT' state:

```
SVRMGR> connect internal
Connected
SVRMGR> startup nomount
ORACLE instance started
```

6. Refer to the SQL Language Reference Guide for the 'CREATE DATABASE' statement syntax - page 4-148.

   Here is a sample create database statement:

```
SVRMGR> create database NEW_NAME
        2> logfile group 1 '$ORACLE_HOME/dbs/log1NEW_NAME.dbf' size 500K,
        3>         group 2 '$ORACLE_HOME/dbs/log2NEW_NAME.dbf' size 500K
        4> datafile '$ORACLE_HOME/dbs/dbsNEW_NAME.dbf' size 20M
        5> maxdatafiles 50;
```

7. Once completed run catalog.sql located in the oracle_home/rdbms/admin directory. This script must be run under the 'SYS' user or connected 'internal'.

   NOTE: If such products as replication, parallel server or procedural option are installed, the following additional scripts must be run:

   ```
   SVRMGR>@oracle_home/rdbms/admin/catalog.sql
   ```

   In addition, if procedural option is installed:

   ```
   SVRMGR>@oracle_home/rdbms/admin/catproc.sql
   ```

   In addition, if replication is installed:

   ```
   SVRMGR>@oracle_home/rdbms/admin/catrep.sql
   ```

   In addition, if parallel server in installed:

   ```
   SVRMGR>@oracle_home/rdbms/admin/catparr.sql
   ```

8. After the database has been created, the SYSTEM tablespace and SYSTEM rollback segment will exist. However, a second rollback segment in the SYSTEM tablespace must be created and activated before any other tablespaces can be created in the database (Refer to SQL Language Reference Manual for full syntax).

   Creating the rollback segment:

   ```
   SYNTAX: CREATE ROLLBACK SEGMENT system2
           TABLESPACE SYSTEM
           STORAGE (...);
   ```

   Activating the rollback segment:

   ```
   SYNTAX: ALTER ROLLBACK SEGMENT system2 ONLINE;
   ```

   You will then need to create a new tablespace. This tablespace should only contain rollback segments. For example, to create a tablespace of size 20M with the name RBS:

   ```
   SVRMGR> create tablespace RBS datafile '<datafile name>' size <20M>;
   ```

   You will then need to create additional rollback segments and bring them online. Once at least one other rollback segment is online, you should then drop the second rollback segment 'system2' created above.

```
SVRMGR> create rollback segment <name1> tablespace RBS size <n>;
SVRMGR> create rollback segment <name2> tablespace RBS size <n>;
SVRMGR> alter rollback segment <name1> online;
SVRMGR> alter rollback segment <name2> online;
SVRMGR> alter rollback segment system2 offline;
SVRMGR> drop rollback segment system2 ;
```

If the rollback segments were created as private rollback segments, you will need to edit the init<SID>.ora parameter file and add the line:

rollback_segments = (<list of rollback segments>)

so that the next time the database is started they are brought online automatically.

9. Run the pupbld.sql script

```
% cd $ORACLE_HOMEsqlplus/admin
% svrmgrl
SVRMGR> connect system/<password>
SVRMGR> @pupbld
```

10. Modify the /etc/oratab file by adding the new database name. This is used by dbstart to startup all databases with a 'Y' entry in this file. (See page 4-17 of Oracle for Unix technical Reference Guide).

Method II - Using the CRDBXXX.SQL Script

Setting the Unix Environment

When creating a database, Oracle looks at the environment to see which database to create. Therefore, it is essential that before creating a database your environment is set with the database name. If you are creating a second database, be sure to use a new database name. The following environment variables need to be set:

ORACLE_SID - set to the database name you wish to create
ORACLE_HOME - set to full pathname of the Oracle system home directory
PATH - needs to include $ORACLE_HOME/bin

To set your Unix environment use the following commands depending on the Unix shell you are using:

```
sh  - ORACLE_SID XXX;export ORACLE_SID
csh - setenv ORACLE_SID XXX
```

Check to be sure it was changed:

```
echo $ORACLE_SID
<NEW_NAME>
```

NOTE: Not changing the ORACLE_SID environment and running the create database may wipe out your existing database and all of its data.

Creating the Database

This method assumes that you have created *at least one database through the install* and therefore have a 'crdbXXX.sql' script (XXX being the created database name). With this option you copy this sql script and make the necessary modifications. This option allows you to make whatever changes are desired, such as the MAXDATAFILES parameter or specifying multiple SYSTEM tablespace database files.

Steps for Method II:

1. Set the UNIX environment (SEE SETTING UNIX ENVIRONMENT SECTION ABOVE).

2. Create new init<sid>.ora and config<sid>.ora files for your new database by copying the default ones provided by Oracle:

   % cp $ORACLE_HOME/dbs/init<sid>.ora $ORACLE_HOME/dbs/initNEW_NAME.ora
   % cp $ORACLE_HOME/dbs/config<sid>.ora $ORACLE_HOME/dbs/configNEW_NAME.ora

3. Change db_name parameter in the new initNEW_NAME.ora from DEFAULT to the new database name and specify the control file's name and location (this is often found in the config<sid>.ora file).

   NOTE: If the database name is not specified when using the create database statement, the name specified for the initialization parameter "DB_NAME" in the init<sid>.ora file is used.

4. Create a new crdbXXX.sql for your new database by copying the existing one in the ORACLE_HOME/dbs directory. NOTE: If you have never created a database through the install, you CANNOT use this method.

5. Modify the crdb<dbname>.sql and change the system datafile, logfiles, database name, and pfile="path/initNEW_NAME.ora"

6. Run the modified create script:

```
$ sqldba lmode=y
SQLDBA> @crdbXXX
```

   or

```
$ svrmgrl
SVRMGR> @crdbXXX
```

   Note: We will use Server Manager in our examples.
   If you are using SQL*DBA (versions prior to V7.3.x),
   the commands are the same.

   OPTIONAL!!! If you wish only to create the system tablespace, skip to step 9. Only go through steps 7 and 8 if you wish to also create the standard tablespaces the install creates.

7. Create a new crdb2<dbname>.sql by coping an existing one.
   Make the necessary changes to is, as above, such as datafile names, sizes,

etc. However, this applies to the standard tablespaces which are created by the install, including RBS, USERS, TEMP, TOOLS.

8. Run the modified script, skip to step 10:

```
SVRMGR> @crdb2XXX
```

9. Run catalog.sql found in the ORACLE_HOME/rdbms/admin directory. This script should be run while connected internal after the database is successfully created.

   NOTE: this step is only necessary if you skipped steps 7 and 8.

```
SVRMGR>@oracle_home/rdbms/admin/catalog.sql
```

10. If such products as replication, parallel server or procedural option are installed, the following additional scripts must be run:

    If procedural option is installed:

```
SVRMGR>@oracle_home/rdbms/admin/catproc.sql
```

    If replication is installed:

```
SVRMGR>@oracle_home/rdbms/admin/catrep.sql
```

    If parallel server in installed:

```
SVRMGR>@oracle_home/rdbms/admin/catparr.sql
```

11. Run the pupbld.sql script

```
% cd $ORACLE_HOMEsqlplus/admin
% svrmgrl
SVRMGR> connect system/<password>
SVRMGR> @pupbld
```

Method III - Using ORAINST

Setting the Unix Environment

When creating a database, Oracle looks at the environment to see which database to create. Therefore, it is essential that before creating a database your environment is set with the database name. If you are creating a second database, be sure to use a new database name. The following environment variables need to be set:

```
ORACLE_SID - set to the database name you wish to create
ORACLE_HOME - set to full pathname of the Oracle system home directory
PATH - needs to include $ORACLE_HOME/bin
```

To set your Unix environment use the following commands depending on the Unix

shell you are using:

```
        sh  - ORACLE_SID XXX;export ORACLE_SID
        csh - setenv ORACLE_SID XXX
```

Check to be sure it was changed:

```
        echo $ORACLE_SID
        <NEW_NAME>
```

NOTE: Not changing the ORACLE_SID environment and running the create database may wipe out your existing database and all of its data.

Creating the Database

Note: This method *cannot* be used for V7.3.2.1 installs, but *can* be used for versions 7.3.2.2 and up.

With this method a database can be created using the orainst. Since this process is menu driven it is easy to use, in addition it will run necessary scripts for any product selected. However, this method does not have all database options available, such as specifying a higher MAXDATAFILES value. In addition, if this method is chosen, you must create all the standard non-system tablespaces.

Here are the steps for Method III:

1. Make a copy of your existing config.ora file in the $ORACLE_HOME/dbs directory. THIS FILE WILL BE OVERWRITTEN WITH THIS METHOD.

   ```
   % cp $ORACLE_HOME/dbs/config.ora $ORACLE_HOME/dbs/configORIG_DB.ora
   ```

2. Run orainst from the $ORACLE_HOME/install directory. (Note: This may also be in the $ORACLE_HOME/orainst directory.)

3. When running orainst a logfile will be generated. Specify a name for this logfile so that this file can be easily found and accessed.

4. From the Install Actions choose:
   'Create New Database Objects'

5. Enter the new SID for this database.

6. From the Select Available Products choose:

   'Oracle7 Server (RDBMS)'

   In addition select all other appropriate products that you will wish to use with the new database. (Note: This option will not reinstall the product software.)

   Once you select all the products, tab to <install> and accept.

7. Continue the installation by answering the appropriate questions.

8. Orainst will let you know what step it is processing. Once it is done, in addition to a database, you will also have a crdb<DBNAME>.sql discussed in Method II above. Be sure to check the log specified in step 2 to be sure no errors occurred.

9. Clear up the config.ora confusion:

   a. Copy config.ora to new database name
      % cp $ORACLE_HOME/dbs/config.ora $ORACLE_HOME/dbs/configNEW_DB.ora
   b. Edit initORIGINAL_DB.ora change the ifile entry from config.ora to configORIG_DB.ora
   c. Edit initNEW_DB.ora change the ifile entry from config.ora to configNEW_DB.ora

---

Method IV - Using the UNIX "CP" Command

Setting the Unix Environment

When creating a database, Oracle looks at the environment to see which database to create. Therefore, it is essential that before creating a database your environment is set with the database name. If you are creating a second database, be sure to use a new database name.
The following environment variables need to be set:

ORACLE_SID - set to the database name you wish to create
ORACLE_HOME - set to full pathname of the Oracle system home directory
PATH - needs to include $ORACLE_HOME/bin

To set your Unix environment use the following commands depending on the Unix

shell you are using:

sh - ORACLE_SID XXX;export ORACLE_SID
csh - setenv ORACLE_SID XXX

Check to be sure it was changed:

echo $ORACLE_SID
<NEW_NAME>

NOTE: Not changing the ORACLE_SID environment and running the create database may wipe out your existing database and all of its data.

Creating the Database

The following procedure describes how to create a second database and instance from an existing database using the UNIX 'cp' command and Oracle's 'create control file statement'. The second instance can also be created by restoring datafiles from a cold backup.

Steps for Method IV:

1. Set the UNIX environment variable to the current database, DatabaseA.
   (SEE SETTING UNIX ENVIRONMENT SECTION ABOVE).

2. Backup the control file to trace From DatabaseA. The trace file
   (ora_xxx.trc) is located in the directory defined by USER_DUMP_DEST.

       SQLDBA>alter database backup controlfile to trace resetlogs;

3. Modify trace file script by doing the following:

   a. Remove the header information.
   b. Modify the LOGFILE to point to the new names of the redo logfiles.
   c. Modify the DATAFILE to point to the new names of the data files.
   d. Modify the create controlfile statement to:
      CREATE CONTROLFILE SET DATABASE DatabaseB RESETLOGS NOARCHIVELOG
      or if the database is in archive log mode:
      CREATE CONTROLFILE SET DATABASE DatabaseB RESETLOGS ARCHIVELOG
   e. Remove everything from script after the end of the
      create controlfile statement.

4. Shutdown DatabaseA. Make sure that this is a NORMAL or IMMEDIATE
   shutdown. DatabaseA MUST NOT be shutdown using SHUTDOWN ABORT.

5. Copy DatabaseA database files and redo logfiles (dbf,log) to the DatabaseB
   directories. The files may alternatively be restored from a cold backup.

6. Copy DatabaseA parameter files to DatabaseB parameter files by
   doing the following:

   a. Copy initDatabaseA.ora to initDatabaseB.ora.
   b. Copy configDatabaseA.ora to configDatabaseB.ora.
   c. Modify initDatabaseB.ora to point to the config.ora of DatabaseB.
   d. Modify the following parameters in initDatabaseB.ora:

          IFILE = configDatabaseB.ora
          CONTROL_FILE = new control files names
          DB_NAME = new database name

      Comment out the parameter REMOTE_LOGIN_PASSWORDFILE if it is set
      to EXCLUSIVE.

      Be sure to copy the files using the same Unix user as the current owner.
      If the files have the wrong permissions, when attempting to create
      the controlfile you are likely to encounter ORA-1503, ORA-1161,
      ORA-1110.

7. Set the following environment variables:

   ORACLE_SID = DatabaseB
   ORACLE_HOME = Full pathname of home directory of DatabaseB

8. Run the trace file script from 3) via SVRMGRL as a privileged user
   (connect internal in Oracle 8.1 or earlier).

After the CREATE CONTROLFILE statement has run, you will need to take one of the following actions:

8a. If this is a copy from a cold backup, issue the command:

```
ALTER DATABASE OPEN RESETLOGS;
```

8b. If the files are from a backup, and you want to roll forward, perform standard media recovery. When the recovery is successfully complete, issue the command:

```
ALTER DATABASE OPEN RESETLOGS;
```

8c. If the files are from an aborted instance, you will need to perform a recovery. The recovery will need to roll forward through the online redo logs copied over with the database. The controlfile will not indicate which online redo log is current. A list of online logs which will indicate which redo log is current can be obtained using the command:

```
SELECT member FROM v$logfile;
```

Issue the following command:

```
RECOVER DATABASE USING BACKUP CONTROLFILE
```

At the recovery prompt apply the online logs in sequence by typing the unquoted full path and file name of the online redo log to apply. After applying the current redo log, you will receive the message 'Media Recovery Complete'. Once the media recovery is complete issue the command:

```
ALTER DATABASE OPEN RESETLOGS;
```

The database is now up and open under the new sid. DatabaseA may be restarted.

```
    alter database rename global_name to DatabaseB.WORLD;
```

9. Modify other utilities:

SQLNET (i.e. listener.ora, tnsnames.ora)
AUTOMATIC STARTUP (i.e. oratab)