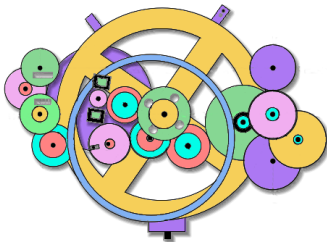
Antikythera Publications



Relational Database Design
http://www.AntikytheraPubs.com
sthalberg @ AntikytheraPubs.com

DATABASE DESIGN NOTE SERIES

Installing Oracle-XE® on 64 bit Ubuntu Operating Systems

Compiled by: "Sig" Thalberg and F. Oberle from a variety of sources

Learning something as complex as relational database design can be greatly accelerated by having one's own personal computer equipped with the same RDBMS used by many of the major corporations the student might aspire to join.

Although the personal license for Oracle-XE (Express Edition) is readily available for a variety of Windows™ operating systems and Unix/Linux distributions, many prefer to use more "user friendly" systems such as those in the popular Ubuntu family for their own personal use.

Oracle doesn't offer its XE RDBMS for Debian-based Linux systems (e.g. Ubuntu), but it isn't that difficult to adapt its RPM package to work with these, and you may learn a bit more about your operating system in the process.

6 June 2014 – Updated November 2016

See page 15 for information on other material from Antikythera Publications.

THIS PAGE IS INTENTIONALLY BLANK

Database Design Note Series – Oracle on 64 bit Ubuntu

Preface – Purpose

Oracle, which provides one of the world's leading Relational Database Management System (RDBMS), offers a free developer license for their product, but *buntu users seem to be left in the cold. All is not lost, however, because in this Design Note we'll show you how to install and run the Oracle 11gR2 Express Edition on 64 bit LTS versions of Ubuntu 12.04 ("Precise Pangolin") through 16.04 ("Xenial Xerus") – this will likely work for other Debian-based systems, but we haven't personally confirmed that. If you are serious about becoming a database guru, this is an easy way to obtain a professional environment in which to do so.

This paper will explain how to obtain the rpm (Red Hat Package Manager) package, convert it to a deb package, perform the initial set up of Oracle XE, and how to begin using Oracle's SQL-Plus command line interface.

There are a few limitations. Oracle is only suitable for 64 bit *buntu installations. The database you can build will be limited to a "mere" 11 GB of user data, and is limited to using only one processor and no more than 1 GB of memory. The installation requires at least 512 MB. Details about Oracle XE can be seen at:

http://www.oracle.com/technetwork/database/database-technologies/express-edition/overview/index.html

To download the package, you will need to create a free on-line account with Oracle – a very small price to pay and, based on our experience, one that doesn't result in any unsolicited material in your in-box. The e-mail and the password you select also gives you access to other developer utilities and resources not discussed in this paper.

Preparation

Prior to installation, you need to confirm that in addition to running on a 64 bit processor, your system is otherwise capable of running the Oracle-XE RDBMS. Throughout these instructions, use of a text editor is required. The examples assume the use of gedit, since that is Ubuntu's default editor, but you may use any text editor[1] you want.

☐ **Step A**: You will need to know the amount of installed RAM as a number of bytes. The easiest way to do this is to see the value displayed in "System Settings ..." from the desktop, but you may also use the command:

```
sudo lshw -short -C memory | grep MHz
```

This is, of course, issued from a command prompt in a shell, opened with Ctrl+Alt+T in Ubuntu. The number must be at least 512 MB; to convert that to an exact byte count, use the formula:

512 * 1,048,576 = 536,870,912 bytes (multiply GB by 1073741824 to obtain bytes)

Write this number down for reference for the later steps. If you wish to allocate more RAM to Oracle, you may do so, but the minimum is usually sufficient for a single user running multiple applications.

☐ **Step B**: Oracle 11gR2 XE requires at least 2GB of available swap space. Determine if your available swap space (in gigabytes) is sufficient by issuing the following command:

```
free -g
```

Look for a line similar to the following:

```
Swap:           13           0          13
```

In this example, the machine has 13 GB available (the third number). If you have insufficient swap space, you will need to either create a swap partition or a swap file. Instructions for doing this in Ubuntu are readily available on the web, and so will not be repeated here.

1 It is assumed that you know enough not to attempt to use a word processor for editing operating system text files.

Pre-Installation Steps

☐ **Step C**: Download the zip file containing the 64 bit Red Hat Linux package named oracle-xe-11.2.0-1.0.x86_64.rpm by selecting downloads from the web page mentioned previously, or by directly choosing:

http://www.oracle.com/technetwork/database/database-technologies/express-edition/downloads/index.html

☐ **Step D**: Once the download is complete, unzip the file in a working directory using the following command:

```
unzip oracle-xe-11.2.0-1.0.x86_64.rpm.zip
```

☐ **Step E**: If you do not have the packages alien, liaio1, and unixodbc, you will need to install them. Since no harm will be done if they are already installed, enter the following command:

```
sudo apt install alien libaio1 unixodbc  #(apt-get on older versions)
```

Package Conversion

☐ **Step F**: Convert the Red Hat .rpm package to an Ubuntu .deb package using the following command:

```
sudo alien --scripts -d oracle-xe-11.2.0-1.0.x86_64.rpm
```

Alien takes quite a while to do all the required conversions and reconfiguring of the rpm package, but you can use this time to perform all of the steps up to "Step M: Perform the Installation," but don't proceed beyond that until a `oracle-xe-11.2.0-1.0.x86_64.deb` or `oracle-xe-11.2.0-2_amd64.deb` package has been created.

What to do while waiting for Alien to complete its conversion

☐ **Step G**: Create a chkconfig script using a text editor:

The Red Hat installer assumes the existence of a file /sbin/chkconfig that doesn't exist by default in Ubuntu. Do not load the chkconfig package that is available for Ubuntu, though, since it can cause errors. Instead, create a new chk-config file directly using the following commands:

```
sudo gedit /sbin/chkconfig
```

Once the blank file (and if it's not blank, something is amiss) opens, copy and paste the following into the editor:

```
#!/bin/bash
# This file was created for, and only required for the
# Oracle RDBMS 11gR2 XE installation.
file=/etc/init.d/oracle-xe
if [[ ! `tail -n1 $file | grep INIT` ]]; then
  echo >> $file
  echo '### BEGIN INIT INFO' >> $file
  echo '# Provides: OracleXE' >> $file
  echo '# Required-Start: $remote_fs $syslog' >> $file
  echo '# Required-Stop: $remote_fs $syslog' >> $file
  echo '# Default-Start: 2 3 4 5' >> $file
  echo '# Default-Stop: 0 1 6' >> $file
  echo '# Short-Description: Oracle 11g XE' >> $file
  echo '### END INIT INFO' >> $file
fi
update-rc.d oracle-xe defaults 80 01
```

Now, save the /sbin/chkconfig file and close the editor.

☐ **Step H**: Now apply execute privileges to the /sbin/chkconfig file you just created using the following command:

```
sudo chmod 755 /sbin/chkconfig
```

You can confirm the settings have been applied properly using the following command:

```
ls -l /sbin
```

You should see something like:

```
-rwxr-xr-x 1 root root 660 Nov 23 20:29 /sbin/chkconfig
```

☐ **Step I**: Define the Linux Kernel parameters

Oracle 11gR2 XE requires the setting of some additional kernel parameters. First we'll create a suitable Oracle configuration file using the editor, which will make the settings permanent by applying them to the kernel so they'll be set on each reboot. First create the file using the editor.

```
sudo gedit /etc/sysctl.d/60-oracle.conf
```

When the blank file is opened, copy and paste the following into the editor:

```
# Oracle 11g XE kernel parameters
# File used to support Oracle RDBMS Installation
fs.file-max=6815744
net.ipv4.ip_local_port_range=9000 65000
kernel.sem=250 32000 100 128
kernel.shmmax=536870912
```

The value used for kernel.shmmax in the last line above (536870912) is the minimum recommended amount of RAM (see **Step A** above). For use of Oracle while learning, this is probably sufficient, but the value may be increased up to the amount of RAM in your system if desired (i.e. the number does not need to match the amount of RAM in your system, but it cannot exceed it.)

Save the sysctl.d/60-oracle.conf file and close the editor.

Confirm the settings have been applied using the following command:

```
sudo cat /etc/sysctl.d/60-oracle.conf
```

☐ **Step J**: Load the Linux Kernel parameters (see /etc/sysctl.d/README for an explanation of this)

```
sudo service procps start       # This Restarts the service
```

This command may return the following if swap is insufficient, but may also return nothing at all:

```
procps stop/waiting
```

Verify the that parameters have been loaded by issuing the command:

```
sudo sysctl -q fs.file-max      # -q is the "quiet" option
```

This command should return something like the following:

```
fs.file-max = 773266
```

☐ **Step K**: On some Ubuntu systems, /dev/shm may be defined as a link to /run/shm, but a link won't satisfy Oracle, so the link needs to be removed (*if it exists*) and replaced with an actual directory which is then mounted. To do so, execute the following commands:

```
sudo rm -rf /dev/shm           # meaning: -recursive -force
sudo mkdir /dev/shm            # shm means Shared Memory
sudo mount -t tmpfs shmfs -o size=2048m /dev/shm
```

(The size value can be any value up to the size of your RAM in MB, but we've found 2048 to be quite sufficient)

To make these changes permanent (*again, only if /dev/shm doesn't already exist*), we need to create another file called S01shm_load in the directory /etc/rc2.d, so another editing session is required:

```
sudo gedit /etc/rc2.d/S01shm_load    # Note the capital "S"
```

Copy and paste the following lines into the editor:

```
#!/bin/sh
# This file was created for and only required by
# the Oracle RDBMS installation.
case "$1" in
  start) mkdir /var/lock/subsys 2>/dev/null
         touch /var/lock/subsys/listener
         rm /dev/shm 2>/dev/null
         mkdir /dev/shm 2>/dev/null
         mount -t tmpfs shmfs -o size=2048m /dev/shm
         ;;
      *) echo error
         exit 1
         ;;
esac
```

Save the file and close the editor.

Confirm the settings have been applied using the following command:

```
ls -l /etc/rc2.d
```

There should be an appropriate line for the new file similar to the following:

```
-rw-r--r-- 1 root root 273 Nov 23 20:41 S01shm_load
```

Now set execute permissions for the file with the command:

```
sudo chmod 755 /etc/rc2.d/S01shm_load
```

Confirm that this was successful by repeating the last command:

```
ls -l /etc/rc2.d
```

If successful, the appropriate result line should look like this:

```
-rwxr-xr-x 1 root root 273 Nov 23 20:41 S01shm_load
```

Using the “mount” command in your shell, confirm that the shmfs temporary file system has been loaded by looking for a line like the following toward the end of the listing:

```
shmfs on /dev/shm type tmpfs (rw,size=2048m)
```

☐ **Step L**: The Red Hat installation process uses the /bin/awk utility but, since Ubuntu places this at /usr/bin/awk, we need to create a symbolic link to the location the Red Hat installer expects by issuing the following command:

```
sudo ln -s /usr/bin/awk /bin/awk
```

Create an empty listener directory and listener file for Oracle’s use by issuing the following commands:

```
sudo mkdir /var/lock/subsys       # This file MAY EXIST ALREADY
sudo touch /var/lock/subsys/listener
```

DO NOT PROCEED BEYOND THIS POINT UNTIL THE `oracle-xe-11.2.0-1.0.x86_64.deb` or `oracle-xe-11.2.0-2_amd64.deb` package has been created and the Alien utility has ended. (see **Step F**).

Installing Oracle

☐ **Step M**: Perform the Installation

Insure that you are in the directory where the .deb file was created in Step F. Begin the installation with the following command:

```
sudo dpkg --install oracle-xe_11.2.0-2_amd64.deb
```

☐ **Step N**: Remove (highlight and delete) or Edit the extraneous Desktop Start Icon.

By default, the Red Hat installation procedure creates a desktop start icon, but this won’t work under Ubuntu, so either reconfigure it later (see “Starting and Stopping Oracle” on page 10), or delete it with the following command:

```
rm $HOME/Desktop/oraclexe-gettingstarted.desktop
```

☐ **Step O**: Configure the Oracle Installation

Once installed, Oracle needs to be configured prior to using it. The utility to do this is run with the following command:

```
sudo /etc/init.d/oracle-xe configure
```

You will be asked to enter the following information:

- A valid HTTP port for the Oracle Application Express (unless you have a good reason for not doing so, accept the default value of 8080 by pressing Enter at the prompt)
- A valid port for the Oracle database listener (unless you have a good reason for not doing so, accept the default value of 1521 by pressing Enter)
- A single password for both the SYS and SYSTEM administrative user accounts; you will be asked to renter the password for confirmation. SAVE THIS PASSWORD!! It is for the user “oracle.”

ORACLE SYSTEM PASSWORDS:	
SYS:	
SYSTEM:	

You will then be asked whether you want the database service to start automatically each time the computer starts. The answer to this will depend on multiple considerations. Having the service start when Ubuntu boots will

lengthen the boot and shutdown process noticeably; on the other hand, starting SQL*Plus to access your database will be instantaneous. So if you wish to have Oracle available quickly during normal operation, and you don't boot your computer very often, choose "Yes." Of course, if your computer is memory constrained[2], this may not be a good option. Instructions for starting and stopping the Oracle processes during operation are given at the end of these installation instructions. Depending on the desktop environment in use, creating icons or menu entries from these commands should be straightforward.

For many Ubuntu users, it may be more convenient to have the service run at log-in, so answer "Yes." You'll see:

```
Starting Oracle Net Listener . . . Done
```

☐ **Step P**: Several environment variables need to be set up for Oracle's SQL*Plus to run in the BASH shell; this is done by editing the existing (hidden) .bashrc file using the following command:

```
sudo gedit $HOME/.bashrc
```

Add the following lines to the end of the .bashrc file (N.B. - the back-tics are not single quotation marks):

```
#=========================================================
# The following lines were added on `date` to
# support the use of Oracle 11gR02 terminal operations
# [SQL*Plus] by [Insert Your Name Here] (or use $USER)
export ORACLE_HOME=/u01/app/oracle/product/11.2.0/xe
export ORACLE_SID=XE
export NLS_LANG=`$ORACLE_HOME/bin/nls_lang.sh`
export ORACLE_BASE=/u01/app/oracle
export LD_LIBRARY_PATH=$ORACLE_HOME/lib:$LD_LIBRARY_PATH
export PATH=$ORACLE_HOME/bin:$PATH
#=========================================================
```

Save the file and close the editor. Reload your profile by issuing the following commands:

```
cd $HOME
. ./.profile
```

The command above is "period, space, period, slash, period" followed by the word "profile." The command will only take a second to execute and won't display anything if it executed successfully.

☐ **Step Q**: Start Oracle:

Depending on whether you chose the option to have Oracle start at boot, the oracle-xe service may or may not have already been started. Execute the following command anyway:

```
sudo service oracle-xe start
```

If the service has already been started, you will receive the message:

```
Oracle Database 11g Express Edition instance is already started
```

Otherwise, the following message will appear:

```
Starting Oracle Database 11g Express Edition instance.
```

2 Since this can only be installed on 64 bit processors, processor speed will generally not be an issue, as even the slowest of these is adequate for running Oracle-XE.

☐ **Step R**: Create a Database User

For normal activities, you should create one or more individual database user accounts. To do so, start SQL*Plus by executing the following command:

```
sqlplus sys as sysdba
```

The following will be displayed, confirming that SQL*Plus has been started and your Oracle-XE installation has been successful:

```
SQL*Plus: Release 11.2.0.2.0 Production on Wed May 9 12:12:16 2012
Copyright (c) 1982, 2011, Oracle. All rights reserved.
Enter password:
```

In this document, the red border is used to indicate that you are actually running the command line version of SQL*Plus. On your screen, you will see no difference between working in your shell and working in Oracle's SQL*Plus environment except for the prompt. Enter the password you chose in **Step O**. Assuming you entered the correct password, the following message will be displayed, confirming that your installation has been successful:

```
Connected to:
Oracle Database 11g Express Edition Release 11.2.0.2.0 - 64bit Production
SQL>
```

It is a good practice to set up one or more separate user accounts for development and experimentation – perhaps even one for each project you are working on. This will permit you to use Oracle without logging out of your current user session in Ubuntu (remember, “Oracle” is a different user.)[3]

ORACLE USER PASSWORDS:	
:	
:	

As an example, you could create a user named JohnQ and set his password to “Jello” (passwords are case-sensitive, although user names are not!) by entering the following command:

```
create user JohnQ identified by Jello;
```

Note, by the way, that Oracle will both require and remind you to change passwords periodically; Oracle-XE is, after all, identical in its behavior to the enterprise version of their RDBMS. The system will respond with:

```
User created.
```

You may then use this login within any Ubuntu user session without the need to log out. In order to give JohnQ the ability to connect to and use the system to create or access a database, enter the following command:

```
grant connect, dba, resource to JohnQ;
```

You may or may not wish to add permissions such as dba to that list depending on your needs, but this will be desirable if you are learning to do development; consult Oracle documentation. The system should respond with:

```
Grant succeeded.
```

Repeat this process to create any other users and suitable “grants” that may be needed at this time. In a home learning environment, it is sometimes helpful to create a user for each project you are working on in order to isolate them from the inevitable errors you'll make while coming to grips with Oracle and SQL.

3 Oracle can also recognize existing operating system user ids and passwords, which may be more convenient in certain circumstances. See Automating User Login in Ubuntu on page 13.

☐ **Step S**: Log in as a User and Confirm Operations

The following command will disconnect you from the SYS account, and then connect you under JohnQ's account to confirm that your user setup was successful:

```
connect JohnQ/Jello
```

If you simply type "sqlplus" alone, SQL*Plus will prompt for a user id and password. A minimal method of confirming that the database is operational is to execute one or both of the following commands:

```
select sysdate from dual;
```

```
select 2 + 2 from dual;
```

The system will return something that looks like the following:

```
SYSDATE
-----------------
5-JUN-14
```

```
2 + 2
-----------------
4
```

As another example of being aware of the differences between the SQL*Plus environment and the BASH environment – if you wanted to repeat the previous SQL command, rather than typing the up arrow and [Enter] key, which is what you would do in BASH, type the [/] key followed by the [Enter] key, which is the closest SQL*Plus equivalent, as it simply reruns the most recent command. Explaining the key commands of either shell is far beyond the scope of this document, but we've found it useful from the standpoint of blood pressure control to set the prompts and colors for one or the other environments to provide some more pronounced visual clue as to the current environment. How to accomplish this in either environment is beyond the scope of this document.

To return to the operating system's BASH terminal environment, simply type "exit" and you'll see your normal command line prompt. Typing "exit" once more will close the BASH shell in the normal manner.

Starting and Stopping Oracle

If you chose not to have Oracle services always available back in **Step O**, the following BASH scripts will give you those capabilities. As mentioned earlier, it is usually possible to turn these into icons or menu entries but, since the process of doing so depends on the desktop environment you are using, is not addressed here.

The following three scripts should be created and then copied to the /usr/local/bin/ directory and given execute permissions (e.g. sudo chmod +x start_oracle). The purpose of the start_oracle and stop_oracle scripts should be self-explanatory. The check_oracle script determines if Oracle services are already running and is used (in the same manner as an "include file") in the first two; you may, of course, call it in the same fashion if you wish to set up more elaborate schemes or scripts.

Start_Oracle Bash Script

```
#!/bin/bash
# start_oracle
# Full Path is /usr/local/bin/start_oracle
# NB:(/usr/local/bin is in Ubuntu's default PATH).
# Frank Oberle; 18 June 2014
# This script starts the Oracle XE service if it is not running.
#
# The script "check_oracle" is run as a . command rather than
# a sub-shell so we can have the value of its $Running variable.
. check_oracle

if [ $Running == 'Y' ]
```

```
then
  echo '========================================================'
  echo '=             Oracle is already Running.               ='
  echo '=      Active Oracle services are listed above.        ='
  echo '========================================================'
  echo '  Press any key to continue ...'
  IFS= read -r -s -n1 -d ''
else
  echo '========================================================'
  echo '=   Starting Oracle Services and Database Instance     ='
  echo '= . .Please wait - this takes a little bit of time. . =='
  echo '========================================================'
  sudo service oracle-xe start
  echo '========================================================'
  echo '= Oracle Services and Database Instance have started  ='
  echo '========================================================'
  echo '  Press any key to continue ...'
  read -r -s -n1 -d ''
fi
```

Stop_Oracle Bash Script

```
#!/bin/bash
# stop_oracle
# Full Path is /usr/local/bin/stop_oracle
# NB: (/usr/local/bin is in Ubuntu's default PATH).
# Frank Oberle; 18 June 2014; modified 2 August 2014
# This runs as expected under both conditions (Oracle running or
not running)
# This script stops the Oracle XE services and database instance
if they are active.
#
# The script “check_oracle” is run as a . command rather than
# a sub-shell so we can have the value of its $Running variable.
. check_oracle

if [ $Running == 'Y' ]
then
  echo '========================================================'
  echo '=    Stopping Oracle Services and Database Instance   ='
  echo '=. . Please wait - this takes a little bit of time . .='
  echo '========================================================'
  sudo service oracle-xe stop
  echo 'Oracle processes have been terminated ...'
  echo 'Press any key to exit.'
  IFS= read -r -s -n1 -d ''
else
  echo '========================================================'
  echo '=       Oracle is not currently running ...            ='
  echo '=     Use "start_oracle" to run the database.          ='
  echo '========================================================'
  echo 'Press any key to exit.'
  IFS= read -r -s -n1 -d ''
fi
```

Check_Oracle Bash Script

```
#!/bin/bash
# check_oracle
# Full Path is /usr/local/bin/check_oracle
# NB: (/usr/local/bin is in Ubuntu's default PATH).
# Frank Oberle; 18 June 2014
# This checks the process listing to see if the Oracle services
# are already running and sets the value of $Running
# appropriately. This is called from within several other
# related Oracle scripts.
#
# We need the "^" (beginning of line character) and trailing
# space in the grep statement as the command "grep oracle" by
# itself will return the process of finding "oracle" (i.e. a
# self reference) and report that an oracle process is
# running when that is misleading ...

if ps -ef | grep '^oracle '
then
  Running=Y
else
  Running=N
fi
```

Example Output – Starting Oracle Services

When running the Start_Oracle script from the command line, for instance, the output will be:

```
============================================================
=      Starting Oracle Services and Database Instance      =
= . . . Please wait - this takes a little bit of time . . . =
============================================================
[sudo] password for your_user_id:
Starting Oracle Net Listener.
Starting Oracle Database 11g Express Edition instance.

============================================================
=  Oracle Services and Database Instance have been started  =
============================================================
  Press any key to continue ...
```

The process stops at the line beginning [sudo] to wait for you to enter your password. This line can be confusing – at this point in the process you are still running in the BASH shell, so Ubuntu is looking for the "super-user (root) password in order to start the Oracle services (owned by root) – not the Oracle password.

At this point, you should be able to begin using Oracle (at least from the command line) to begin building and populating tables, and testing various SQL, SQL*Plus, and PL/SQL capabilities.

Eventually, you may want to explore the wide variety of interfaces to support various front-ends and database connectivity options, as well as graphical management and development tools, but these are too numerous to even consider covering in a paper of this sort. The internet is your friend should you decide to proceed further, but our recommendation is to first become familiar with the fundamentals using this existing setup, since graphical tools often hide some of the things you will need to know if you want to live in this world.

Automating User Login in Ubuntu

As with any Unix system, Oracle can be set up to recognize a user from their existing Ubuntu user id. To do this, first log in to SQL*Plus using the Oracle SYSTEM user ID (see page 7). Then issue the following commands:

```
CREATE USER OPS$OS-User-Name IDENTIFIED EXTERNALLY ;
GRANT CONNECT, RESOURCE, DBA TO OPS$OS-User-Name ;
EXIT ;
```

The *OS-User-Name* should of course be whatever operating system user name you are defining. Once this has been done, all that will be required to log in to SQL*Plus is to type "`sqlplus /`".

Enhancing the command line version of SQL*Plus

One drawback of using the command line version of SQL*Plus is that, even when running under the Bash Shell, there is no support for the usual editing aids Bash provides. The most important of these is the ability to use the up and down arrow keys to retrieve recent commands. Installing a Linux support utility called rlwrap will provide such support and is recommended for any serious use of SQL*Plus. Fortunately, rlwrap is available in most repositories, and can easily be installed with the following command:

```
sudo apt install rlwrap    # apt-get depending on Linux version
```

This can be tested by exiting SQL*plus if it is active, and then entering the following at the Bash prompt.

```
alias rsql='rlwrap sqlplus'
```

This will then permit the command "rsql" to start SQL*Plus; you can of course use any alias you like, son long as it doesn't conflict with an existing command. Enter a few of the commands shown earlier in **Step S**, and then using the up arrow key to return to the earlier commands. Once the functionality has been confirmed, the change may be made permanent by adding the alias command as shown above to the .bashrc file as described in **Step P**.

Uninstalling Oracle Software

To completely uninstall the Oracle RDBMS, the following procedure, which is essentially undoing all the commands listed above, begins by executing the following commands from a shell or terminal:

```
sudo -s
/etc/init.d/oracle-xe stop
sudo ps -ef | grep oracle | grep -v grep | awk '{print $2}' |
xargs kill
sudo dpkg --purge oracle-xe
sudo rm -r /u01
sudo rm /etc/default/oracle-xe
sudo update-rc.d -f oracle-xe remove
```

Remove the following operating system files that are no longer needed:

```
sudo rm /sbin/chkconfig
sudo rm /etc/sysctl.d/60-oracle.conf  (may not exist)
sudo rm /etc/rc2.d/S01shm_load    # IF you created it in Step K
```

Remove the lines previously added to the end of $HOME/.bashrc in **Step P**. This will prevent any warning messages about non-existent files from appearing when starting a shell.

Remove the symbolic link to awk:

```
sudo ln -s /usr/bin/awk /bin/awk
```

```
```

Empty and then remove the /var/lock/subsys directory:

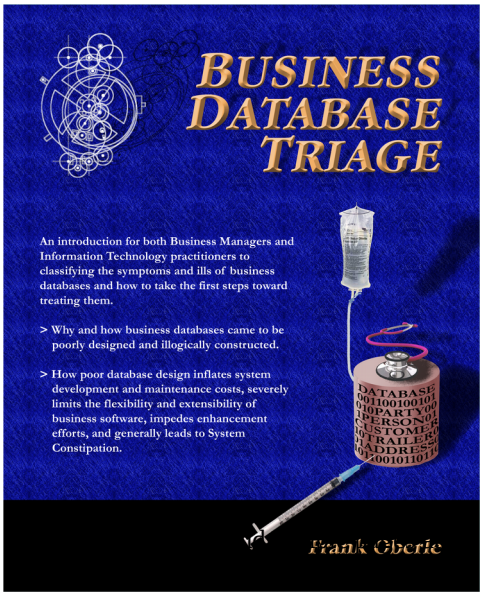
```
sudo rm -Ir /var/lock/subsys
sudo rmdir /var/lock/subsys
```

Removing the Database(s)

The Oracle system software has now been removed, but the database files themselves still remain in the /u01/app/oracle/ subdirectory. These may be required later or moved to another system. The database files are owned by the “oracle” operating system user, so you must log in to Ubuntu under that user or as root if you wish to remove the files and their directories permanently.

Other Publications

Antikythera Publications



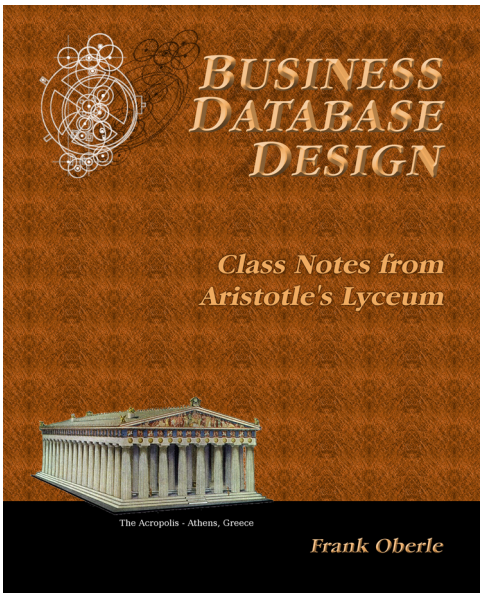
In addition to an ongoing series of Database Design Notes, Antikythera Publications recently released the book "*Business Database Triage*" (ISBN-10: 0615916937) that demonstrates how commonly encountered business database designs often cause significant, although largely unrecognized, difficulties with the development and maintenance of application software. Examples in the book illustrate how some typical database designs impede the ability of software developers to respond to new business opportunities – a key requirement of most businesses.

A number of examples of solutions to curing business system constipation are presented. Urban legends, such as the so-called object-relational impedance mismatch, are debunked – shown to be based mostly on illogical database (and sometimes object) designs.

"*Business Database Triage*" is available through major book retailers in most countries, or from the following on-line vendors, each of which has a full description of the book on their site:

CreateSpace: https://www.createspace.com/4513537

Amazon:
www.amazon.com/Business-Database-Triage-Frank-Oberle/dp/0615916937



A follow-up book, "*Business Database Design – Class Notes from Aristotle's Lyceum*" is due to be available in the early part of 2015.

"*Business Database Design*" leads the reader through the logical design and analysis techniques of data organization in more detail than the earlier work – which concentrated more on understanding and identifying problems caused by illogical database design rather than their solutions.

These logical approaches to data organization, espoused by Aristotle and an "A-List" of his successors, have formed the basis for scientific discovery over more than 2,400 years, and directly led to the technology we deal with today, notably including both relational and object theory.

"*Business Database Triage*" explained the reasons why these principles were virtually impossible to apply during the early years of our transition to the use of computers in business, but since the technology is now sufficiently mature that such compromises can no longer be justified, the time has come to relearn logical data organization techniques and apply them to our businesses.

To download the ERD_A TrueType Font used in this document, along with a tutorial and keyboard map, visit www.AntikytheraPubs.com, where a variety of other Database Design Notes are also available for download.